MAINVIEW® Batch Optimizer Job Optimizer Reference Manual

Version 2.3

July 31, 2003

bmcsoftware

Copyright 2003 BMC Software, Inc. All rights reserved.

BMC Software, the BMC Software logos, and all other BMC Software product or service names are registered trademarks or trademarks of BMC Software, Inc. All other trademarks belong to their respective companies.

BMC Software considers information included in this documentation to be proprietary and confidential. Your use of this information is subject to the terms and conditions of the applicable End User License Agreement for the product and the proprietary and restricted rights notices included in this documentation.

Restricted Rights Legend

U.S. Government Restricted Rights to Computer Software. UNPUBLISHED -- RIGHTS RESERVED UNDER THE COPYRIGHT LAWS OF THE UNITED STATES. Use, duplication, or disclosure of any data and computer software by the U.S. Government is subject to restrictions, as applicable, set forth in FAR Section 52.227-14, DFARS 252.227-7013, DFARS 252.227-7014, DFARS 252.227-7015, and DFARS 252.227-7025, as amended from time to time. Contractor/Manufacturer is BMC Software, Inc., 2101 CityWest Blvd., Houston, TX 77042-2827, USA. Any contract notices should be sent to this address.

Contacting BMC Software

You can access the BMC Software Web site at **http://www.bmc.com**. From this Web site, you can obtain information about the company, its products, corporate offices, special events, and career opportunities.

United States and Canada

Address	BMC Software, Inc. 2101 CityWest Blvd. Houston TX 77042-2827
Telephone	713 918 8800 or 800 841 2031
Fax	713 918 8000

Outside United States and Canada

Telephone	(01) 713 918 8800
Fax	(01) 713 918 8000

Customer Support

You can obtain technical support by using the Support page on the BMC Software Web site or by contacting Customer Support by telephone or e-mail. To expedite your inquiry, please see “Before Contacting BMC Software.”

Support Web Site

You can obtain technical support from BMC Software 24 hours a day, 7 days a week at **http://www.bmc.com/support_home**. From this Web site, you can

- read overviews about support services and programs that BMC Software offers
- find the most current information about BMC Software products
- search a database for problems similar to yours and possible solutions
- order or download product documentation
- report a problem or ask a question
- subscribe to receive e-mail notices when new product versions are released
- find worldwide BMC Software support center locations and contact information, including e-mail addresses, fax numbers, and telephone numbers

Support by Telephone or E-mail

In the United States and Canada, if you need technical support and do not have access to the Web, call 800 537 1813. Outside the United States and Canada, please contact your local support center for assistance. To find telephone and e-mail contact information for the BMC Software support center that services your location, refer to the Contact Customer Support section of the Support page on the BMC Software Web site at **http://www.bmc.com/support_home**.

Before Contacting BMC Software

Before you contact BMC Software, have the following information available so that Customer Support can begin working on your problem immediately:

- product information
  - product name
  - product version (release number)
  - license number and password (trial or permanent)
- operating system and environment information
  - machine type
  - operating system type, version, and service pack or other maintenance level such as PUT or PTF
  - system hardware configuration
  - serial numbers
  - related software (database, application, and communication) including type, version, and service pack or maintenance level
- sequence of events leading to the problem
- commands and options that you used
- messages received (and the time and date that you received them)
  - product error messages
  - messages from the operating system, such as `file system full`
  - messages from related software

Contents

Chapter 6 Job Optimizer Commands

Chapter 7 Job Transformation Language

Chapter 8 Job Optimizer Pipes Introduction

Chapter 9 **Job Optimizer Pipes Dialog**

Chapter 10 **Defining Job Optimizer Pipes Initialization Parameters**

Chapter 11

Job Optimizer Pipes Operator Commands

Chapter 12

Job Optimizer Pipes Implementation Considerations

Chapter 13 **Job Optimizer for DB2**

Chapter 14 **Job Optimizer for IMS**

Appendix A Job Optimizer Reports

Appendix B Navigating the User Interface

Tables

BMC Software, Inc., Confidential and Proprietary Information

Figures

BMC Software, Inc., Confidential and Proprietary Information

About This Book

This book contains detailed information about the Job Optimizer component of MAINVIEW® Batch Optimizer.

To use this book, you should be familiar with the following items:

- Multiple Virtual Storage (MVS) systems, job control language (JCL), and the Interactive System Productivity Facility (ISPF)
- your client and host operating systems

For example, you should know how to respond to ISPF panels.

How This Book Is Organized

This book is organized as follows:

Chapter/Appendix	Description
Chapter 1, “Introduction”	explains the features and benefits of Job Optimizer
Chapter 2, “Understanding Job Optimizer”	explains what Job Optimizer is, what it does, and how it works
Chapter 3, “Implementing Job Optimizer”	describes the tasks necessary to implement Job Optimizer
Chapter 4, “Defining Job Policies”	describes how to view or modify Job Policy definitions
Chapter 5, “Defining the User Control Facility”	describes how to view or modify the User Control Facility
Chapter 6, “Job Optimizer Commands”	lists Job Optimizer commands
Chapter 7, “Job Transformation Language”	explains Job Transformation Language
Chapter 8, “Job Optimizer Pipes Introduction”	explains the features and benefits of the Job Optimizer Pipes component
Chapter 9, “Job Optimizer Pipes Dialog”	explains how to use the Job Optimizer Pipes Component
Chapter 10, “Defining Job Optimizer Pipes Initialization Parameters”	explains how to define Job Optimizer Pipes initialization parameter members
Chapter 11, “Job Optimizer Pipes Operator Commands”	describes the Job Optimizer Pipes commands
Chapter 12, “Job Optimizer Pipes Implementation Considerations”	explains implementation considerations of Job Optimizer Pipes
Chapter 13, “Job Optimizer for DB2”	explains how to use Job Optimizer in a DB2 environment
Chapter 14, “Job Optimizer for IMS”	explains how to use Job Optimizer in an IMS environment
Appendix A, “Job Optimizer Reports”	explains the reports that can be generated by Job Optimizer
Appendix B, “Navigating the User Interface”	explains how to use the product interface
Appendix C, “User Control Facility Internal Policy”	describes the User Control Facility internal policy
Appendix D, “Job Optimizer for DB2 and SQL Statements”	describes DB2 and SQL statements
Appendix E, “Migrating from BatchPipes to Job Optimizer Pipes”	explains the Job Optimizer Pipes rules and the BatchPipes parameters that are supported by MAINVIEW Batch Optimizer
Appendix F, “IOA and INCONTROL User Considerations”	explains the interface between the IOA components and MAINVIEW Batch Optimizer
Appendix G, “CA-11 Considerations”	explains the steps needed to disable CA-11 tracking for steps taken in the Job Optimizer component

Related Documentation

BMC Software products are supported by several types of documentation:

- online and printed books
- online Help
- release notes and other notices

Note: The messages that the MAINVIEW Batch Optimizer dialog generates are available in an MVS data set that is downloaded during installation. For each message, the data set includes an explanation and suggests a user response.The MVS data set is called *HLQ*.BSS.ISPMLIB (where *HLQ* is the high-level qualifier that is specified during installation).

In addition to this book and the online Help, you can find useful information in the publications listed in the following table. As “Online and Printed Books” on page xxi explains, these publications are available on request from BMC Software.

Document	Description
MAINVIEW Batch Optimizer Data Optimizer Reference Manual	describes the Data Optimizer component of MAINVIEW Batch Optimizer
MAINVIEW Batch Optimizer Messages Manual	lists and describes the messages that are generated by MAINVIEW Batch Optimizer components
MAINVIEW Batch Optimizer General Information	describes MAINVIEW Batch Optimizer
MAINVIEW Batch Optimizer Installation Guide	provides information and instructions for installing MAINVIEW Batch Optimizer
BMC Software Subsystem Manual	provides information and instructions for operating the BMC Software Subsystem

Online and Printed Books

The books that accompany BMC Software products are available in online format and printed format. If you are a Windows or Unix user, you can view online books with Acrobat Reader from Adobe Systems. The reader is provided at no cost, as explained in “To Access Online Books.” You can also obtain additional printed books from BMC Software, as explained in “To Request Additional Printed Books.”

To Access Online Books

Online books are formatted as Portable Document Format (PDF) files. You can view them, print them, or copy them to your computer by using Acrobat Reader 3.0 or later.You can access online books from the documentation compact disc (CD) that accompanies your product or from the World Wide Web.

In some cases, installation of Acrobat Reader and downloading the online books is an optional part of the product-installation process. For information about downloading the free reader from the Web, go to the Adobe Systems site at **http://www.adobe.com**.

To view any online book that BMC Software offers, visit the support page of the BMC Software Web site at **http://www.bmc.com/support_home**. Log on and select a product to access the related documentation. (To log on, first-time users can request a user name and password by registering at the support page or by contacting a BMC Software sales representative.)

To Request Additional Printed Books

BMC Software provides printed books with your product order. To request additional books, go to **http://www.bmc.com/support_home.**

Online Help

MAINVIEW Batch Optimizer includes online Help. In the MAINVIEW Batch Optimizer ISPF interface, you can access Help by pressing **F1** from any ISPF panel.

Release Notes and Other Notices

Printed release notes accompany each BMC Software product. Release notes provide current information such as

- updates to the installation instructions
- last-minute product information

In addition, BMC Software sometimes provides updated product information between releases (in the form of a flash or a technical bulletin, for example). The latest versions of the release notes and other notices are available on the Web at **http://www.bmc.com/support_home**.

Conventions

This section describes the conventions that are used in this book.

This section provides examples of the conventions that are used in this book and explains how to read ISPF syntax statements.

General Conventions

This book uses the following general conventions:

Item	Example
information that you are instructed to type	Type **SEARCH DB** in the designated field. Type **search db** in the designated field. (Unix)
specific (standard) keyboard key names	Press **Enter**.
field names, text on a panel	Type **the appropriate entry** in the **Command** field.
directories, file names, Web addresses	The BMC Software home page is at **www.bmc.com**.
nonspecific key names, option names	Use the HELP function key. KEEPDICTIONARY option
MVS calls, commands, control statements, keywords, parameters, reserved words	Use the SEARCH command to find a particular object. The product generates the SQL TABLE statement next.

code examples, syntax statements, system messages, screen text	`//STEPLIB DD`
	`The table` *table_name* `is not available.`

emphasized words, new terms, variables	The instructions that you give to the software are called *commands*. In this message, the variable *file_name* represents the file that caused the error.
single-step procedures	» To enable incremental backups, type **y** and press **Enter** at the next prompt.

This book uses the following types of special text:

Note: Notes contain important information that you should consider.

Warning! Warnings alert you to situations that could cause problems, such as loss of data, if you do not follow instructions carefully.

Syntax Statements

Syntax statements appear in Courier. The following example shows a sample syntax statement:

```
COMMAND KEYWORD1 [KEYWORD2|KEYWORD3] KEYWORD4={YES|NO}
       file_name...
```

The following table explains conventions for syntax statements and provides examples:

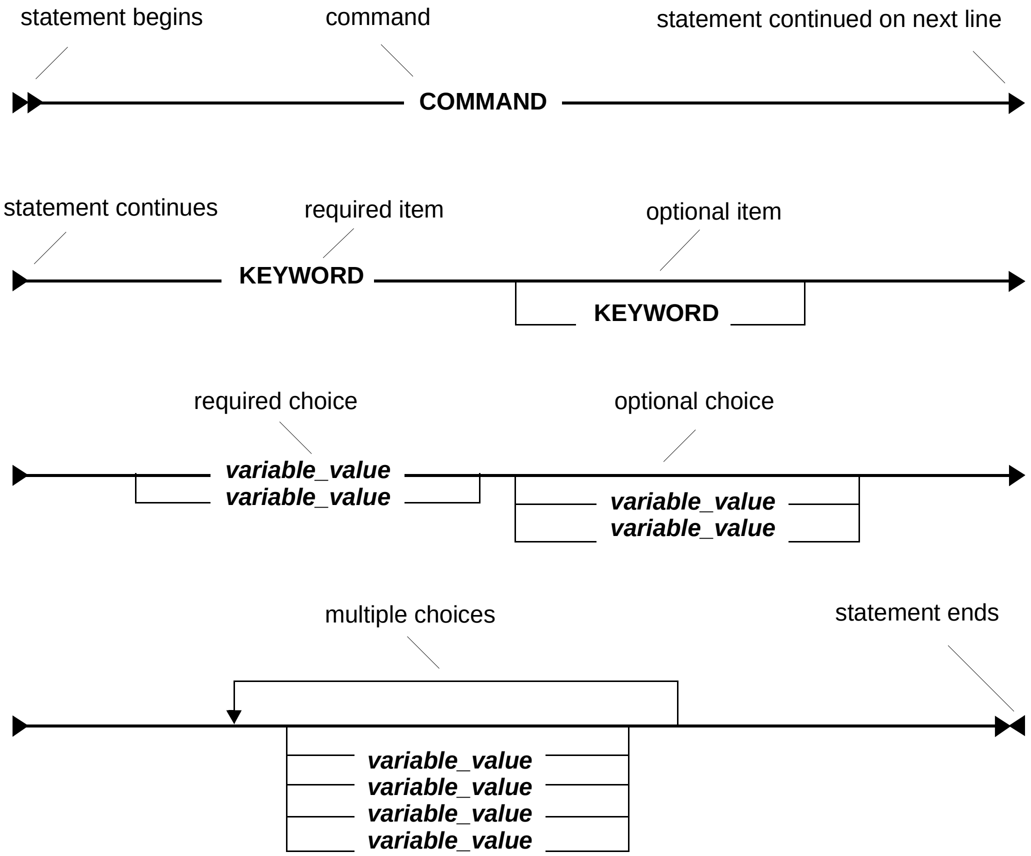
Item	Example
Items in italic type represent variables that you must replace with a name or value.	`dtsbackup` *control_directory*
Brackets indicate a group of options. You can choose at least one of the items in the group, but none of them is required. A comma means that you can choose one or more of the listed options. You must use a comma to separate the options if you choose more than one option.	[*table_name, column_name, field*]
Braces enclose a list of required items. You must enter at least one of the items.	{*DBD_name* | *table_name*}

A vertical bar means that you can choose only one of the listed items. In the example, you would choose either *commit* or *cancel*.	`{commit | cancel}`
	`{-commit | -cancel} (Unix)`

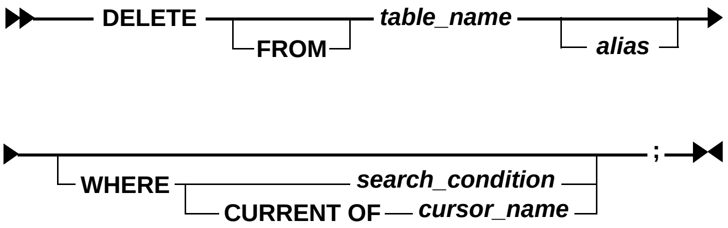
An ellipsis indicates that you can repeat the previous item or items as many times as necessary.	*column_name* . . .

Syntax Diagrams

The following figure shows the standard format for syntax diagrams:



The following example illustrates the syntax for a DELETE statement. Because the FROM keyword, *alias* variable, and WHERE clause are optional, they appear below the main command line. In contrast, the *table_name* variable appears on the command line because the table name is required. If the statement includes a WHERE clause, the clause must contain either a search condition or a CURRENT OF clause. (The *search_condition* variable appears on the main line for the WHERE clause, indicating that this choice is required.)



The following guidelines provide additional information about syntax diagrams:

- A recursive (left-pointing) arrow above a stack indicates that you may choose more than one item in the stack.

- An underlined item is a default option.

- In general, MVS commands, keywords, clauses, and data types appear in uppercase. However, if an item can be shortened, the minimum portion of the MVS command or keyword may appear in uppercase with the remainder of the word in lowercase (for example, CANcel).

- The following conventions apply to variables in syntax diagrams:

  — Variables typically appear in lowercase and are always italicized.

  — If a variable is represented by two or more words, underscores connect the words (for example, *database_name* and *user_ID*).

Chapter 1 Introduction

This chapter explains the Job Optimizer and Job Optimizer Pipes components of MAINVIEW® Batch Optimizer. This chapter discusses the following topics:

MAINVIEW Batch Optimizer Overview

The MAINVIEW Batch Optimizer product is a specialized set of components that manage batch jobs, reduce elapsed processing times, and provide more efficient use of your available resources. Some MAINVIEW Batch Optimizer components provide common core functionality to the product, while others provide optimization for specific types of performance processing.

MAINVIEW Batch Optimizer is available in three product tiers, each providing a different level of functionality to meet your batch processing optimization needs:

- MAINVIEW Batch Optimizer–Standard
- MAINVIEW Batch Optimizer–Advanced
- MAINVIEW Batch Optimizer–Enterprise

Table 1-1 lists the components that are included with each MAINVIEW Batch Optimizer product tier.

Table 1-1 Components of MAINVIEW Batch Optimizer by Product Tiers

Included component	MAINVIEW Batch Optimizer–Standard	MAINVIEW Batch Optimizer–Advanced	MAINVIEW Batch Optimizer–Enterprise
Job Optimizer	no	yes	yes
Job Optimizer Pipes	no	yes	yes
Job Optimizer for DB2 and IMS	no	no	yes
Data Optimizer	yes	yes	yes

Job Optimizer

Job Optimizer provides performance benefits to batch jobs by running job steps concurrently and providing step-to-step piping to enable the movement of data between the split steps.

Job Optimizer is provided with MAINVIEW Batch Optimizer–Advanced and MAINVIEW Batch Optimizer–Enterprise.

Job Optimizer Pipes

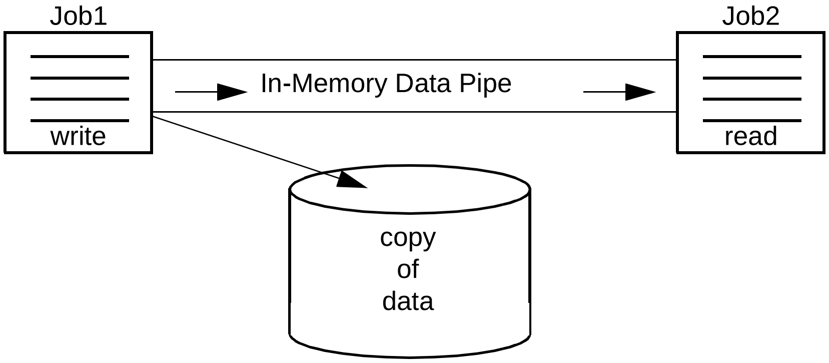
Job Optimizer Pipes provides in-memory piping of application data between batch jobs or job steps. By using pipes two data-related applications can execute concurrently rather than sequentially, reducing the elapsed time required to process the jobs.

Job Optimizer Pipes establishes an in-memory pipe between one (or more) jobs that creates the data, writers, and one (or more) jobs that will read the data, readers. In-memory pipes between job steps running in parallel are established by Job Optimizer. Using a pipe, the reader can access a block as soon as it is written. The writer and reader jobs or steps can execute on the same image or on another image in the Parallel Sysplex.

Data passed through a pipe exists only until it is read by the readers. In situations where a physical copy of the data is required for future use or for backup purposes, Job Optimizer Pipes allows the data that passed through the pipe to be written to a physical file on DASD or tape. This function provides the ability to execute the writer and reader jobs concurrently, while still preserving a hardened copy of the data.

Figure 1-1 shows two data-dependent jobs executing in parallel, with the data being piped through memory and concurrently copied to disk.

Figure 1-1 Job Optimizer Pipes Component



Job Optimizer Pipes is provided with MAINVIEW Batch Optimizer–Advanced and MAINVIEW Batch Optimizer–Enterprise.

Job Optimizer for DB2 and IMS

Job Optimizer for DB2 and IMS provides performance benefits to batch jobs that access DB2 and IMS databases. Performance is optimized by running job steps concurrently.

Job Optimizer for DB2 and IMS offers the same performance benefits as Job Optimizer; however, Job Optimizer for DB2 and IMS executes job steps in parallel that access DB2 or IMS databases.

Job Optimizer for DB2 and IMS is provided only with MAINVIEW Batch Optimizer–Enterprise.

Data Optimizer

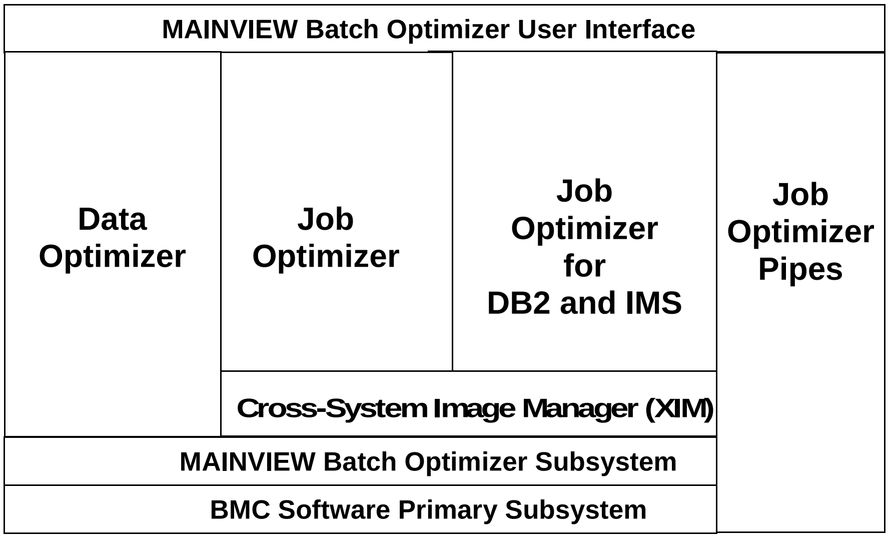
Data Optimizer provides I/O performance benefits by applying a robust set of optimization techniques to VSAM and non-VSAM I/O processing.

Data Optimizer comprises MAINVIEW Batch Optimizer–Standard and is included with the Advanced and Enterprise product packages.

MAINVIEW Batch Optimizer Common Components

Figure 1-2 shows the core components and optimization components of MAINVIEW Batch Optimizer.

Figure 1-2 MAINVIEW Batch Optimizer Components


MAINVIEW Batch Optimizer User Interface
Data Optimizer
Job Optimizer
Job Optimizer for DB2 and IMS
Job Optimizer Pipes
Cross-System Image Manager (XIM)
MAINVIEW Batch Optimizer Subsystem
BMC Software Primary Subsystem

This manual describes functionality that is specific to the Job Optimizer, Job Optimizer Pipes, and Job Optimizer for DB2 and IMS components of the MAINVIEW Batch Optimizer.

Several components provide core functionality to the MAINVIEW Batch Optimizer product.

MAINVIEW Batch Optimizer User Interface

The MAINVIEW Batch Optimizer user interface (dialog) is an ISPF-based interactive dialog that provides access to the MAINVIEW Batch Optimizer control data set. You can use the ISPF Edit function to view or modify the control data set members. The MAINVIEW Batch Optimizer user interface affords a protected environment for this process. The dialog verifies that any control data set members which you create or modify are syntactically correct and do not provide conflicting instructions to the optimization components.

The dialog provides control for all portions of the product and is included with all tiers of MAINVIEW Batch Optimizer.

MAINVIEW Batch Optimizer Subsystem

MAINVIEW Batch Optimizer Subsystem (MBOS) is responsible for providing the following key services to the other components:

- console communication
- history data set access

- common service routines

The MBOS is included with all tiers of MAINVIEW Batch Optimizer.

BMC Software Primary Subsystem

BMC Software Primary Subsystem (BMCP) is responsible for establishing interception points so that MAINVIEW Batch Optimizer components get control at the required times. The BMCP is included with all tiers of MAINVIEW Batch Optimizer.

Cross-System Image Manager (XIM™)

Job Optimizer incorporates the BMC Software Cross-System Image Manager (XIM™) technology. XIM enables Job Optimizer to distribute and manage job steps across one or more MVS systems. XIM functions transparently within Job Optimizer, but it requires the presence of the IBM Cross-System Coupling Facility (XCF).

To permit the distribution of job steps across multiple operating system images, XCF must be executing in a multisystem environment.

Performance Processing with Job Optimizer and Job Optimizer Pipes

Job Optimizer and Job Optimizer Pipes address the major problems facing businesses with batch workloads: lengthy processing time, contention for resources, and unbalanced workloads. Job Optimizer provides your I/S staff with an integrated suite of functions to solve these problems.

Job Optimizer uses parallelism, workload distribution, step-to-step piping, and Shared Record Positioning (SRP) to reduce your company's batch processing time and balance the workload across your system or Parallel Sysplex. By running job steps in parallel, Job Optimizer can dramatically reduce the elapsed time of batch jobs and job streams. Further elapsed time reductions result when you use Job Optimizer to automatically route job steps to available OS/390 images in the Parallel Sysplex.

Job Optimizer Pipes uses job-to-job pipes to enable parallel processing of jobs, and step-to-steps pipes to allow steps that are running in parallel to share data. By running jobs and steps in parallel using pipes, Job Optimizer Pipes can reduce I/O operations and increase batch processing efficiency. Further elapsed time reduction can be gained by running the jobs or steps sharing the pipe in available OS/390 images in the Parallel Sysplex.

With Job Optimizer and Job Optimizer Pipes, your organization's jobs can run faster and process larger volumes of data in less time. Your images are freed to run more work, perhaps extending the time that interactive applications are available or supporting your company's work in another time zone. The resulting increase in data availability and decrease in batch processing time translate directly into business value.

Parallelism

Job Optimizer uses parallelism, which reduces the elapsed time of multi-step batch jobs. For each such job, Job Optimizer analyzes the execution characteristics of the job's steps during normal sequential processing. Based on this analysis, and if no constraints exist, Job Optimizer can split the job steps into separate units of work. Job Optimizer can then run the split job steps in parallel on the same image, or separate images, during subsequent runs of the job.

Job Optimizer automatically applies its step parallelism functions to batch jobs that meet user-specified criteria and whose history of job step behavior indicate that the steps can be parallelized.

Workload Distribution

To best use the resources in your Parallel Sysplex, Job Optimizer can target work to another image whose processing capabilities are more suited to running the job step. Job Optimizer uses workload management (WLM) provided performance data to help it decide where to target work.

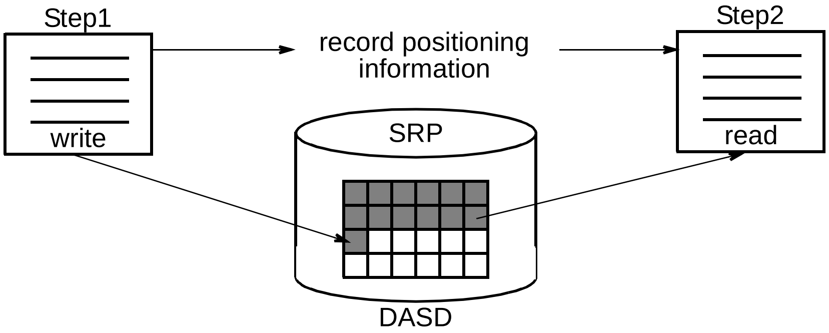
Job Optimizer provides a mechanism for you to override Job Optimizer's decisions to target a job step.

Shared Record Positioning

Job Optimizer can use Shared Record Positioning (SRP) when a data set that resides on DASD has to be shared between a writer/reader pair of steps running parallel. SRP allows data blocks that have already been written to disk by the writer step to be read by subsequent job steps running in parallel with the writer step, before the current writer step completes. Job Optimizer periodically passes the location of the last written block on DASD to the reader step, letting the reader step know how much data is available to be read. The writer and reader steps can run on the same or different systems in the Parallel Sysplex.

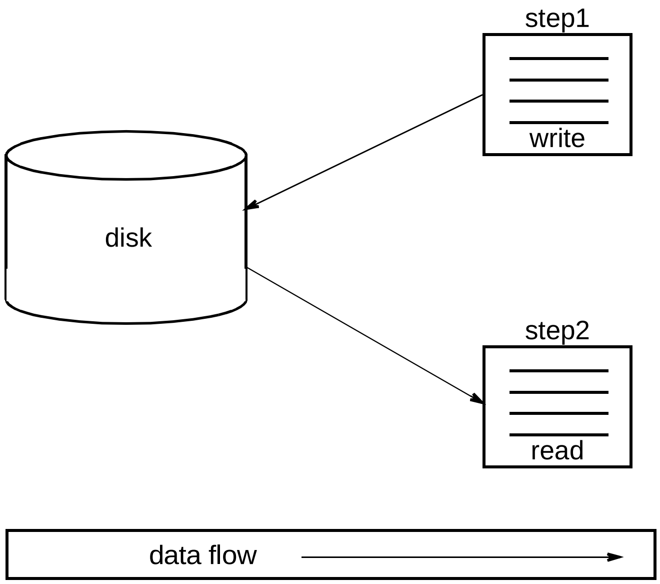
Consider the example shown in Figure 1-3. Step1 processes data that is read by Step2. As Step1 writes data to DASD, Job Optimizer periodically passes the location of the last written record on DASD to Step2. Step2 can read into the data set only as far as Step1 has written.

Figure 1-3 Shared Record Positioning



In Figure 1-4 on page 1-9, shows a normal sequence of steps processing. Step1 writes data to Disk. When all the records have been written, the data set is closed and Step1 terminates. Step2 then reads the data from Disk. When Step2 finishes processing all of the records, the data set is no longer required and can be deleted. In effect, the transfer of data from one step to the other is at a data set level.

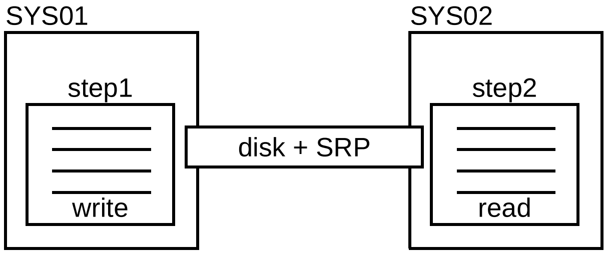
Figure 1-4 Two Steps in a Traditional Batch Job Stream



Compare the processing of the two steps in Figure 1-4 with Figure 1-5, which shows those same two steps using Job Optimizer. The steps are executed in different systems. SRP allows step2 to read blocks as they are written by step1.

In Figure 1-5, step1 (a writer to the dataset) and step2 (a reader from the dataset) run concurrently. Step2 can obtain data from the data set when step1 writes the first block. Output from step1 becomes immediately available as input to step2.

Figure 1-5 Two Steps running in parallel by using Job Optimizer and SRP



Step-to-Step Piping

Job Optimizer can use step-to-step piping through Job Optimizer Pipes when data sharing between steps running in parallel is required. In step-to-step piping, a pipe forms between two data dependent steps, allowing them to execute in parallel passing the data from the writer step to the reader step through the pipe. Using a pipe, the reader can access a block as soon as it is written.

Job-to-Job Piping

Job Optimizer Pipes has the ability to perform job-to-job piping. Job-to-job piping allows two data dependent jobs to execute in parallel. A data pipe forms between the writer step in one job and the reader step in another job, allowing the reader job to access a block as soon as it is written.

Features

Job Optimizer provides the following features:

- concurrent job step execution of batch jobs

  Job Optimizer executes job steps in parallel to minimize the job's elapsed time. Job Optimizer performs parallel execution of job steps automatically.

- concurrent job step execution of DB2 and IMS batch jobs

  Job Optimizer executes DB2 and IMS job steps in parallel to minimize the job's elapsed time. Job Optimizer performs parallel execution of DB2 and IMS job steps automatically.

- concurrent job step execution in Parallel Sysplex environments

  Job Optimizer executes job steps in parallel on any system in the sysplex.

Job Optimizer Pipes provides the following feature:

- in-memory pipes

  Job Optimizer Pipes allows data-related jobs or job steps to transfer data while running in parallel, without using I/O resources.

Benefits

Job Optimizer provides the following benefits:

- reduces elapsed time for execution of all batch jobs
- batch cycle time reductions provide improved online availability
- dynamic workload balancing

Job Optimizer Pipes provides the following benefits:

- reduces the elapsed time for execution of batch jobs and job streams
- reduces the number of physical I/O operations
- reduces contention for I/O resources
- reduces tape and DASD usages
- provides more efficient use of CPU time

BMC Software, Inc., Confidential and Proprietary Information

BMC Software, Inc., Confidential and Proprietary Information

Chapter 2 Understanding Job Optimizer

This chapter describes what Job Optimizer does and how it works. This chapter discusses the following topics:

BMC Software, Inc., Confidential and Proprietary Information

BatchPlex

Job Optimizer uses an entity known as the BatchPlex to manage and improve performance processing for your sysplex. A BatchPlex is required to manage the copies of Job Optimizer that are installed and active on the various images in the sysplex. The BatchPlex requires access to a history data set and a control data set.

Job Optimizer uses the history data set to collect historical information regarding the jobs that run in your installation. History data is used by Job Optimizer to determine which jobs are to be split and how they can be split.

The control data set contains job policies, user control facility (UCF) policies, commands and definitions that identify the images that comprise the BatchPlex.

The MAINVIEW® Batch Optimizer components operate cooperatively within a BatchPlex. A BatchPlex is a group of subsystems on one or more images in a sysplex that provide the following functions:

- parallel processing of job steps
- data transfer through Shared Record Positioning (SRP)
- I/O performance processing with Data Optimizer

You can have more than one BatchPlex in your installation. Job Optimizer balances the batch workload across your BatchPlex and reduces your batch processing window.

Job Optimizer lets you design policies to achieve your site's goals of improving batch control, relieving resource contention, and balancing the workload. In policies, you specify the batch jobs that Job Optimizer is to process, and the options that Job Optimizer is to use with this processing.

BatchPlex Definitions

BatchPlex definitions identify the MVS images to which Job Optimizer can direct job steps for processing and on which Data Optimizer can provide I/O performance processing. Each BatchPlex definition contains the following characteristics:

- policy definition pointers, which identify names of the job policy or data policy to be activated at BatchPlex initialization

- global options, which identify information applicable to all images in the BatchPlex (unless overridden by an MVS image definition)

- MVS image definitions, which identify image-specific overrides to the BatchPlex global options

Job Policy Definitions

Job policy definitions associate one or more batch job characteristics with a Job Optimizer response. Each job policy definition contains the following characteristics

- selection criteria, which identify batch jobs based on characteristics such as job name, execution class, accounting field, or user ID and which identify the name of the action definition to be applied to these jobs

- global options, which identify information applicable to all selection definitions in the policy (unless overridden by an action definition)

- action definitions, which identify the performance options that should be used for a given selection definition

UCF Definitions

UCF policy definitions associate specific installation conditions with a Job Optimizer response. Each UCF policy definition

- provides installation information for esoteric names
- allows specific actions, such as split or bypass, to be based on program or DD names

Figure 2-1 on page 2-4 shows the basic structures of the definitions that are stored in a MAINVIEW Batch Optimizer control data set.

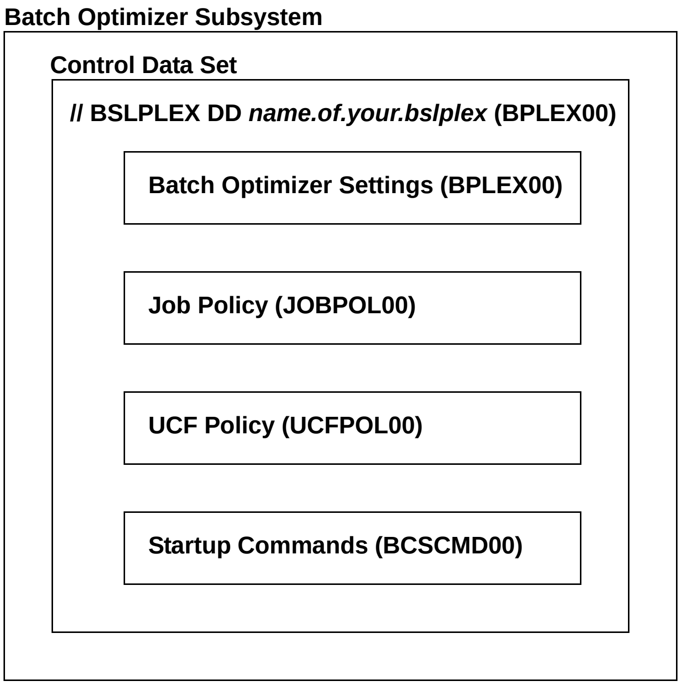
BMC Software, Inc., Confidential and Proprietary Information

Commands

The commands member identifies commands that will be executed during subsystem initialization:

- initialize Job Optimizer
- issue any MVS commands
- define JES3 initiator groups

Figure 2-1 Control Data Set Definitions - Basic Structures


Batch Optimizer Subsystem
Control Data Set
// BSLPLEX DD *name.of.your.bslplex* (BPLEX00)
Batch Optimizer Settings (BPLEX00)
Job Policy (JOBPOL00)
UCF Policy (UCFPOL00)
Startup Commands (BCSCMD00)

For more information about the BatchPlex, see Chapter 5, “Defining the User Control Facility.”

Job Policies

A job policy is a collection of selection criteria and performance options for batch jobs.

When a batch job is initiated, Job Optimizer checks the policy to determine whether the batch job matches the criteria that is specified in a selection definition. If the criteria matches, Job Optimizer intercepts the job and performs the action specified in the action definition.

Job Optimizer scans a policy from the first definition to the last definition. Therefore, the order in which you place definitions in a policy is important. Place the definitions with the most restrictive selection criteria at the top of the list, and save the least restrictive (or most general) descriptions for the bottom of the list.

If the batch job does not match the criteria in a definition, Job Optimizer takes no action. The job is processed as normal.

When you create policies (using the MAINVIEW Batch Optimizer user interface), you save them as members of the control data set. You can place multiple job policies in the control data set, but only one job policy is active at a time. You can add new policies to the control data set and activate them at any time.

It might make sense for your site to maintain different policies for different batch windows. For example, you can use one set of policies for prime shift batch windows and another set of policies for off–shift batch windows.

As you create your policies, remember that Job Optimizer searches the active job policy for selection and action definitions. The more definitions you have, the longer it takes to perform a search. Keep search time to a minimum by:

- using generic job name specifications whenever possible
- keeping the number of definitions in a policy to approximately 100

For more information about job policies, see Chapter 4, “Defining Job Policies.”

User Control Facility

When a job matches a job policy selection, Job Optimizer interrogates the UCF for each program and DD name to determine if any action is required. UCF can override the job policy selection and bypass the job. UCF can force a job step to execute on a particular image, or to execute in the Job Entry Subsystem (JES) initiator. For example, the UCF can be used to perform such tasks as targeting all FOCUS programs to a specific image.

Job Optimizer cannot share virtual (VIO) data sets. If a step uses VIO and that data set is passed to another step, then all steps involved with the VIO data set must run in the JES initiator. However, UCF can be used to redirect VIO data sets to DASD. The redirection to DASD will allow Job Optimizer to split the steps.

Job Optimizer at times needs to know if an esoteric unit name is disk or tape. Job Optimizer usually recognizes esoteric names defined during system initialization. However, if your site uses products which allow dynamic definition of esoteric names, it may be necessary to add UCF entries so that Job Optimizer recognizes the unit name.

Example

DEVICE = DEVXXX
DEVTYPE = DISK

or

DEVICE = T3480
DEVTYPE = TAPE

Where DEVICE is equal to esoteric name and DEVTYPE is equal to disk or tape.

For more information about the UCF, see Chapter 5, “Defining the User Control Facility.”

History Data

Job Optimizer stores its analysis data in a VSAM cluster known as the history file. The history file is pointed to by the REGISET DD in the MAINVIEW Batch Optimizer subsystem.

When a job matches a job policy other than BYPASS, Job Optimizer will query the history file to determine if it has gathered history for that particular job. After the job terminates, Job Optimizer will update the history file with the analysis data.

Defining IEFACTRT to SMFPRMxx

The Job Optimizer uses an IEFACTRT exit routine to collect historical information about jobs that run during installation. Job Optimizer uses this information to determine which jobs are to be split and how to split them.

You must ensure that the IEFACTRT exit point is correctly defined in member SMFPRMxx of PARMLIB. Job Optimizer requires that its IEFACTRT exit routine receives control for all batch jobs that run during installation. Therefore, ensure that IEFACTRT is specified on the SYS parameter and any of the SUBSYS parameters for which the EXITS option is specified. Doing so will ensure that the IEFACTRT exit routine is called for all batch jobs. For OS/390 and higher environments, and all MAINVIEW Batch Optimizer supported releases, ensuring that IEFACTRT is defined in SMFPRMxx is all that is required on your part for this exit. During initialization MAINVIEW Batch Optimizer automatically adds the IEFACTRT exit to the list of exits that are invoked at the system level for batch processing.

Subsystems

Job Optimizer uses three subsystems to perform job splitting. Each of the following subsystems must be defined to MVS:

- BMC Software Primary Subsystem (BMCP)
- MAINVIEW Batch Optimizer subsystem (MBOS)
- Extended job execution subsystem (XJS)

The BMC Software Primary Subsystem provides operating system level services for Job Optimizer. There may be only one BMC Software Primary Subsystem running on an image. The BMC Software Primary Subsystem is designed to provide services to as many BMC Software products as necessary.

The MAINVIEW Batch Optimizer subsystem is a private copy of the BMC Software Consolidated subsystem (BCSS). The MAINVIEW Batch Optimizer subsystem provides Job Optimizer specific requirements such as

- access to the history file
- MVS command support
- various OS/390 functions

Because the MAINVIEW Batch Optimizer subsystem is a private copy of the BMC Software Consolidated subsystem, MAINVIEW Batch Optimizer cannot share it’s subsystem with other BMC Software products. Job Optimizer and Data Optimizer require the exclusive use of the MAINVIEW Batch Optimizer subsystem.

The extended job execution subsystem is a subsystem to manage XJS subsystem initiators. The XJS handles the functions required to initiate work within an XJS initiator.

Performance Processing for Batch Jobs

For batch jobs, Job Optimizer responds in one of the following ways:

- bypasses the job, allowing normal sequential processing
- analyzes the job during normal sequential processing
- splits the job steps for parallel processing, targets job steps to images, and pipes data between steps to enable data-dependent processing
- reads JTL (job transformation language), which are Job Optimizer control cards within the job that provide user-directed batch performance processing

When you request Job Optimizer to bypass a job, the job processes as it would without Job Optimizer. Analysis and splitting are not performed. You would use bypass processing to restrict specific jobs when a general solution is used, for example, if you were selecting jobs based on a wildcard representation. If the job does not match any selection criteria, Job Optimizer bypasses the job and provides no performance processing.

When you request analysis of a job, Job Optimizer does not act on the job, but merely observes the job and records its observations. Job Optimizer studies the job structure—the pattern of EXEC and DD statements. Job Optimizer watches the data activity as each job step executes and determines which steps access each data set and the nature of each access (such as read, write, create, or modify). Job Optimizer observes CPU utilization during step execution. Job Optimizer saves all analysis data (job history) in the history data set. Stable application behavior is required for Job Optimizer to safely automate split processing. Splitting does not occur if history is not available for the job.

For jobs that have been analyzed sufficiently, when you request job splitting, Job Optimizer performs job performance processing. Job Optimizer determines what steps can be run in parallel, and what steps are more CPU or I/O intensive. Job Optimizer targets the appropriate job steps for execution on the available images, intelligently balancing the batch work load across the sysplex. Job Optimizer pipes data between split steps and merges the sysout output and the split step job logs into a single job where the job was initiated.

Read Job Transformation Language (JTL) commands are Job Optimizer control cards within a job. These commands provide user directed job performance processing. Job Optimizer performs the action specified by JTL. For more information about JTL, see Chapter 7, “Job Transformation Language.”

Information Used to Determine Whether a Job Can Be Split

Job Optimizer records information in the history data set for an eligible job. The following information is used to determine whether a job can be split:

- job structure (the pattern of EXEC and DD statements)
- data access counters
- database access
- CPU utilization during step execution
- data activity as each job step executes

  Job Optimizer can determine how steps access data sets and the nature of each access (input, output, and update).

Determining Whether a Job Can Be Split

When Job Optimizer determines that a job is eligible for splitting, the program takes the following steps to determine whether the job can be split:

Note: The actions in steps 2, 3, and 4 are completed in the Job Optimizer subsystem before execution of the first step.

1. Job Optimizer builds a model of the job's structure. The model contains the EXEC and DD statements of the job.

2. Job Optimizer searches for information about the job in the history data set. Job Optimizer determines whether the current job has any history.

   If there is no history, Job Optimizer cannot split the job. Job Optimizer does not continue with these steps, but it analyzes the job.

   If there is history, Job Optimizer continues to step 3.

3. Job Optimizer determines whether the job's current structure matches what is in the history data set.

   A job matches if its current structure matches the structure in its history. It has the same number of job steps and the same program names.

   If there is no match, Job Optimizer cannot split the job. The program does not continue with these steps, but it does analyze the job.

   If there is a match, Job Optimizer continues to step 4.

4. Job Optimizer performs the following tasks:

- merges the job's current information with its history information
- increments the appropriate data access counters

5. Job Optimizer compares the data access counters to the stability counter. You set the value of the stability counter in the job policy.

   If the stability counter is less than the largest value of any data access counter, Job Optimizer cannot relieve the data constraints. Job Optimizer does not continue with these steps, but it does analyze the job.

   If you specify in the job policy to reset the data access counters to zero, the counter is reset.

   If you do not specify that the data access counters be reset to zero, the counter values remain as set in step 4. The specification is based on the number and order of the DD statements in the job.

   If the stability counter equals or exceeds the value of any data access counter, Job Optimizer continues to step 6.

6. User applies JTL and/or UCF policy options.

7. Job Optimizer determines if the job has any splitting constraints. A job can have data constraints (see “About Data Constraints” on page 2-10), step constraints (see “About Step Constraints” on page 2-13), or both. Data constraints are examined first. If there are no data constraints, Job Optimizer determines if there are step constraints.

If there are no splitting constraints, Job Optimizer splits the job steps.

If there are splitting constraints, Job Optimizer:

- attempts to relieve data constraints as data sets are opened
- relieves step constraints as job steps complete

About Data Constraints

Job Optimizer assumes a data constraint exists when two steps access the same data set.

To maintain integrity, all data constraints must be relieved before job steps can be executed in parallel.

Relievable Data Constraints

Job Optimizer assumes that the data constraint for a job can be relieved if one of the steps writes to the data set and the other one reads from the data set or if both steps just read the data set. The actions taken to facilitate executing the steps in parallel are different in the following cases:

- For reader-reader pairs (two steps that read the same data set), Job Optimizer overrides any exclusive enqueue that might prevent shared access. Since only read access is required, the steps can be scheduled for parallel execution, and each step can read the existing data set according to its program.

- For writer-reader pairs (one step writes to a data set and the other step reads from the data set), Job Optimizer inserts a pipe between the output of the writer step and the input of the reader step. Then Job Optimizer overrides the exclusive enqueue that might prevent shared access and schedules the steps for parallel execution.

Non-Relievable Data Constraints

Job Optimizer cannot make any assumptions or attempt to relieve data constraints for any of the following situations:

- writer-writer pairs (two steps both write to a common data set)

- reader-writer pairs (one step reads from the data set and the other step writes to the data set)

- dynamically allocated data sets with conflicting access types (SHR/EXCL or EXCL/EXCL)

- VSAM data sets with DISP = OLD

- BDAM direct access data sets

- ISAM data sets

- dynamically allocated data sets that are also allocated in the JCL and either or both are DISP=NEW, DISP=OLD, or DISP=MOD

- other direct or keyed access data sets, including Direct, Keyed, or EXCP access method

Overriding Job Optimizer Assumptions

When you are certain that data access patterns do not compromise data integrity, you may insert JTL statements, which direct Job Optimizer to relieve certain data constraints. Job Optimizer will take the JTL statements into consideration when attempting to schedule and target steps for parallel execution.

Warning! JTL overrides the internal logic of Job Optimizer. If JTL is misused, abends or data integrity issues could occur. For example, two programs could write to the same file, simultaneously.

BMC Software, Inc., Confidential and Proprietary Information

About Step Constraints

Job Optimizer recognizes various splitting constraints. Job Optimizer does not split a step if any of the following constraints exist:

- data set usage:
  - The step uses a dynamically allocated data set that has no history. You can override this constraint with JTL.
  - The step uses a data set that is dynamically allocated and allocated in JCL and one of them is not set to SHR. You can override this constraint with JTL.
  - The step uses a temporary SMS data set with DISP=MOD.
  - The step uses a VIO data set, and DISP=(,DELETE) is not specified for the VIO DD statement. You can override this constraint with JTL.
  - The step uses a tape.
- conditional processing:
  - The step is associated with a job-level condition code associated with it. You can override this constraint when defining job policies or with JTL.
  - The step has a step-level condition code and the named step has not completed. If conditional execution does not name a step, all previous steps must complete before the step is split. You can override this constraint when defining job policies, by using JTL, or by updating the history data set.
  - The step contains IF-THEN-ELSE JCL statements and the named step has not completed. If conditional execution does not name a step, all previous steps must complete before the step is split. You can override this constraint when defining job policies, by using JTL, or by updating the history data set.
- referback usage:
  - A PGM=*.nnnnnn referback exists on the EXEC statement.
  - The step uses referback logic that Job Optimizer cannot resolve.

- additional constraints:

  - The step uses an internal reader.
  - The step has unresolved GDG references.
  - The step has too many DD statements; the maximum size of the internal text file could be exceeded.
  - The step never completed normally. You can override this constraint with JTL or by updating the history data set.
  - You specified a JTL statement that prevents the step from splitting.
  - You specified a UCF-member keyword that prevents the step from splitting.
  - The step was restarted from a checkpoint.
  - The step is associated with an unresolved forward reference.
  - A requested target system is not available.
  - The step accesses DB2 and you are not executing Job Optimizer for DB2.
  - The step accesses IMS and you are not executing Job Optimizer for IMS.
  - The step contains a data set that Job Optimizer Pipes will convert to a pipe, as well as, write to the physical data set.

Job Optimizer issues step summary messages (if step summary messages have been requested) that indicate the reasons steps were not split.

Chapter 3 Implementing Job Optimizer

This chapter describes the tasks that are required for implementation of Job Optimizer. This chapter discusses the following topics:

Implementation Strategies

Job Optimizer can optimize job performance automatically to provide job performance improvements across a BatchPlex, but you might want to focus its efforts on a subset of your batch workload.

The production control staff should review the planning tasks outlined in this chapter before implementing Job Optimizer to provide job performance processing to selected batch jobs.

Before you use Job Optimizer you should perform the following tasks:

- select an appropriate implementation strategy for your site
- plan selection criteria for use in policy definitions
- plan job optimization options for use in job policy definitions
- set the global values for performance options that remain fairly unchanged for your site

You can choose from a variety of implementation strategies for Job Optimizer. Your strategy selection will be based partly on available resources, partly on your need for job performance improvements, and partly on your experience with the product.

Initial Strategy

Initially, you will probably run Job Optimizer in test mode, limiting its exposure in your site but planning for an expanded scope in the future. To accomplish this strategy, you can begin by creating two definition statements.

Set the action option of one statement to SPLIT. Set the other action option to ANALYZE.

Specify selection criteria that will encompass some test jobs. The following jobs are prime candidates for test mode:

- jobs that pass sequential data between steps
- jobs with steps that are independent

When a test job is run for the first time, Job Optimizer analyzes the job rather than attempting to split it. Running a test job allows Job Optimizer to observe the job structure and data access patterns of the job and record this information in the history data set.

On subsequent executions of the same job, if Job Optimizer has sufficiently analyzed the job, Job Optimizer applies job performance processing to the job.

This criteria pair would add minimal overhead to the initial strategy and would save implementation time for the intermediate strategy. For more information, see “Intermediate Strategy.”

You would probably start the BatchPlex manually on one MVS image, as your system programmer did for the final customization steps. The job policy global options would have the action option set to BYPASS (the default) as a reminder that no other jobs will be analyzed or split.

Intermediate Strategy

As you gain experience with Job Optimizer, you will probably want to increase the job performance improvements in your site and use a more automated approach. To automate processing, you need to modify some initial definition statements in your job policy criteria and add more definition statements.

Definition selection criteria can also be expanded for other jobs that are to be split as part of a more advanced implementation strategy, such as jobs that include CPU-intensive steps, sort steps, or data extraction steps.

You will probably want to extend the BatchPlex to multiple MVS images, and you might want to start the BatchPlex more automatically. Your system programmer can help identify appropriate images and evaluate various methods of starting the subsystems that are associated with a BatchPlex.

Advanced Strategy

Eventually, you will need to carry out a more systematic approach to letting Job Optimizer balance the sysplex workload. It is now time to identify entire categories of jobs for splitting (jobs in a job class, for example, instead of specific job names). You might also extend the BatchPlex to include all MVS images (if you did not already do so as part of your intermediate strategy).

Choosing Candidates for Job Optimizer

Job Optimizer can provide automatic job performance processing because it can select candidates. Job Optimizer chooses candidates based on job policy information and job structure.

You can define a job policy that covers a group of jobs and Job Optimizer will begin analysis. If the job meets Job Optimizer’s criteria, performance processing is applied to subsequent runs.

A better approach is to use your existing System Management Facilities (SMF) data as a starting point for Job Optimizer’s analysis. Job Optimizer provides a utility that analyzes SMF data and creates an output file (the Select File). Job Optimizer also provides a reporting program that uses that select file and creates a Candidate Report.

The Candidate report will list jobs that may be enhanced by Job Optimizer. SMF does not contain all the necessary information to guarantee candidacy, but it does provide a good starting point.

When a set of candidates have been identified by using SMF data, you can use the Populate History utility to load the Select File to your MAINVIEW Batch Optimizer subsystem history file. You can specify a subset of these jobs during the load process.

Method 1 – Automatic Candidate Selection

Step 1 Define the job policy to ANALYZE|SPLIT a group of jobs.

Step 2 Wait for the job to execute.

Step 3 If ACTION=ANALYZE was selected, Job Optimizer will continue to analyze until you change the ACTION. Running the report program will assist you in determining what benefit might be derived from Job Optimizer.

Step 4 If ACTION=SPLIT was selected, Job Optimizer will automatically enhance the job, if possible, when the trust factor is met.

Method 2 – Candidate Selection Using SMF Data

Step 1 Tailor sample BSLBJ*RPT* to create a Select File, and use it as input to the candidate report program.

Step 2 Analyze the results of the candidate report program to determine whether you want to make more runs to refine the Select File.

Step 3 When you are satisfied with the Select File, tailor BSLBJLOD to populate the MBOS history file.

Step 4 Make sure job policy definitions are in place to cause selection.

Step 5 When the policy is in place, and the trust factor is met, Job Optimizer will apply job performance processing on the next execution.

BMC Software, Inc., Confidential and Proprietary Information

BMC Software, Inc., Confidential and Proprietary Information

Chapter 4 Defining Job Policies

This chapter describes how to view or modify job policy definitions online. Including information about the panels and dialog actions that you use to accomplish these tasks. This chapter discusses the following topics:

BMC Software, Inc., Confidential and Proprietary Information

Understanding Job Policy Definitions

Job policy definitions associate one or more batch job characteristics with a Job Optimizer response. Each job policy definition contains the following characteristics:

- selection criteria, which identify batch jobs based on characteristics such as job name, execution class, accounting field, or user ID and which identify the name of the action definition to be applied to these jobs
- global options, which identify information applicable to all selection definitions in the policy (unless overridden by an action definition)
- action definitions, which identify the performance options that should be used for a given selection definition

Registering Job Policy Definitions

Job policy definitions associate one or more batch job characteristics with a Job Optimizer response. This section describes dialog panels and how you can use them to perform job policy registration tasks.

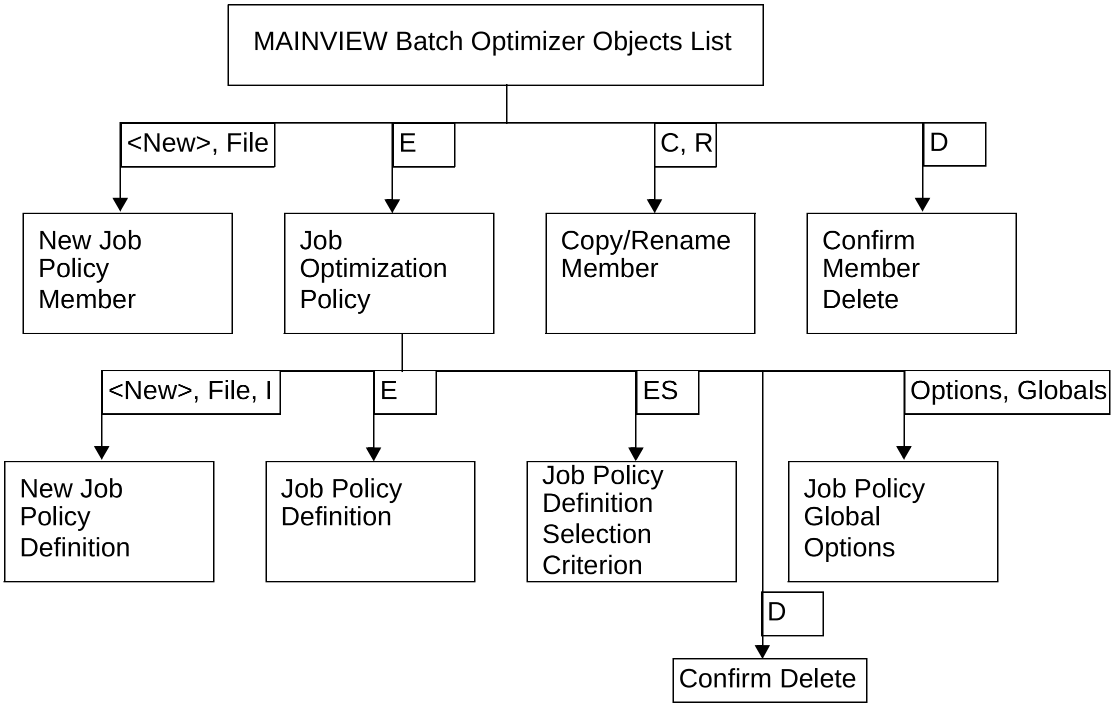
Help is available online for all panels, pop-ups, and fields while registering job policy definitions.

During MAINVIEW Batch Optimizer customization, a sample job policy definition is created in the control data set. The panels and pop-ups shown in this section reflect that sample job policy.

Panel Flow for a Job Policy Definition

Figure 4-1 on page 4-3 shows the basic panel flow in the job policy registration area of the MAINVIEW Batch Optimizer customer interface (dialog). The figure indicates the action code you type (E, ES, I, C, R, D) or the pull-down option you select (File) to move from one panel or pop-up to another.

Figure 4-1 Job Policy Registration—Dialog Panel Flow



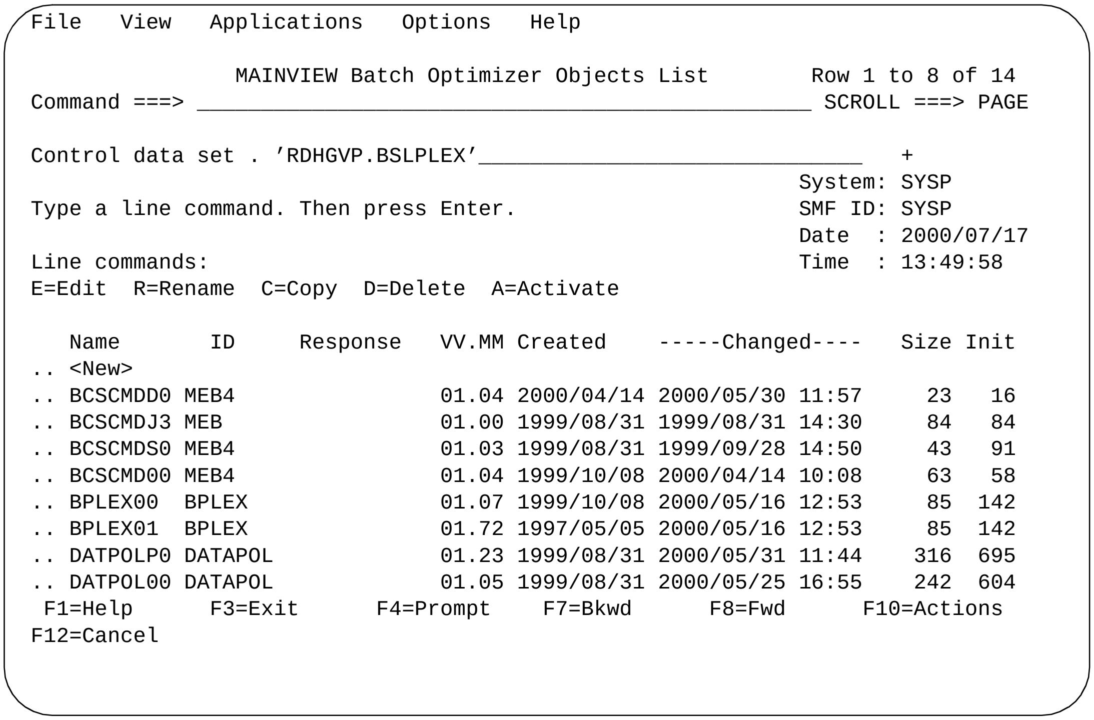
Editing or Viewing a Job Policy

You can review or modify the information saved for a job policy, including the action and definitions associated with the policy.

To edit or view information for a job policy in the control data set, complete the following steps:

Step 1 Access the MAINVIEW Batch Optimizer Objects List panel (Figure 4-2).

Figure 4-2 MAINVIEW Batch Optimizer Objects List Panel



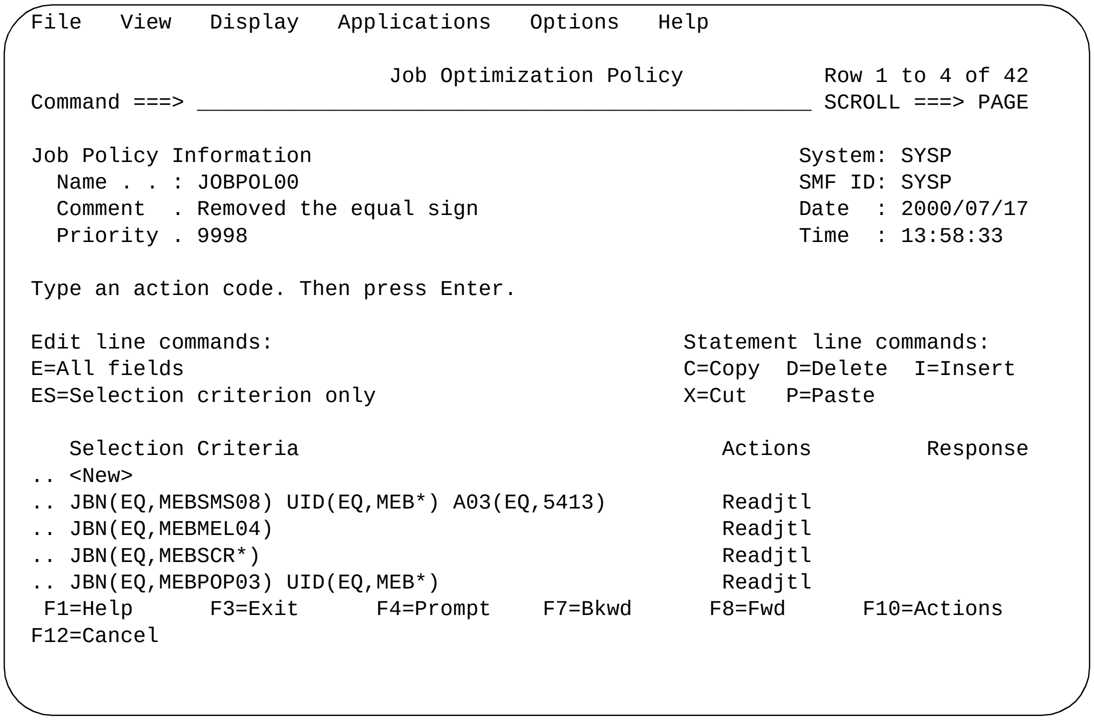
```
File   View   Applications   Options   Help

                MAINVIEW Batch Optimizer Objects List        Row 1 to 8 of 14
Command ===> ________________________________________________ SCROLL ===> PAGE

Control data set . ’RDHGVP.BSLPLEX’______________________________   +
                                                            System: SYSP
Type a line command. Then press Enter.                      SMF ID: SYSP
                                                            Date  : 2000/07/17
Line commands:                                              Time  : 13:49:58
E=Edit  R=Rename  C=Copy  D=Delete  A=Activate

   Name       ID     Response   VV.MM Created    -----Changed----   Size Init
.. <New>
.. BCSCMDD0 MEB4                01.04 2000/04/14 2000/05/30 11:57     23   16
.. BCSCMDJ3 MEB                 01.00 1999/08/31 1999/08/31 14:30     84   84
.. BCSCMDS0 MEB4                01.03 1999/08/31 1999/09/28 14:50     43   91
.. BCSCMD00 MEB4                01.04 1999/10/08 2000/04/14 10:08     63   58
.. BPLEX00  BPLEX               01.07 1999/10/08 2000/05/16 12:53     85  142
.. BPLEX01  BPLEX               01.72 1997/05/05 2000/05/16 12:53     85  142
.. DATPOLP0 DATAPOL             01.23 1999/08/31 2000/05/31 11:44    316  695
.. DATPOL00 DATAPOL             01.05 1999/08/31 2000/05/25 16:55    242  604
 F1=Help      F3=Exit      F4=Prompt    F7=Bkwd      F8=Fwd      F10=Actions
F12=Cancel
```

Step 2 Position the cursor in the action entry field to the left of your choice of job policy. Type **E** (Edit), and press **Enter**.

The dialog displays the Job Optimization Policy panel, which lists the selection criteria and actions in the job policy. The selection criteria are listed first, followed by all actions. Within each definition type, the list reflects your choice of definition order. For each definition, the list indicates the associated definition pointer. When that field is blank, the job policy global options apply to that definition.

Figure 4-3 shows an example of the Job Policy Definition panel, where four selection criteria and four actions are listed. This sample job policy was created during the customization process. You can use this job policy for practice as you learn to manipulate job policies by using the MAINVIEW Batch Optimizer dialog. You can edit this job policy and tailor it to the needs of your site, or you can delete this job policy and create a new one.

Figure 4-3 Job Optimization Policy Panel



```
File   View   Display   Applications   Options   Help

                            Job Optimization Policy           Row 1 to 4 of 42
Command ===> ________________________________________________ SCROLL ===> PAGE

Job Policy Information                                      System: SYSP
  Name . . : JOBPOL00                                       SMF ID: SYSP
  Comment  . Removed the equal sign                         Date  : 2000/07/17
  Priority . 9998                                           Time  : 13:58:33

Type an action code. Then press Enter.

Edit line commands:                                Statement line commands:
E=All fields                                       C=Copy  D=Delete  I=Insert
ES=Selection criterion only                        X=Cut   P=Paste

   Selection Criteria                                 Actions         Response
.. <New>
.. JBN(EQ,MEBSMS08) UID(EQ,MEB*) A03(EQ,5413)         Readjtl
.. JBN(EQ,MEBMEL04)                                   Readjtl
.. JBN(EQ,MEBSCR*)                                    Readjtl
.. JBN(EQ,MEBPOP03) UID(EQ,MEB*)                      Readjtl
 F1=Help      F3=Exit      F4=Prompt    F7=Bkwd      F8=Fwd      F10=Actions
F12=Cancel
```

The Job Optimization Policy panel is a table display and includes a Row field. The example in Figure 4-3 shows this panel as if displayed on a screen large enough to display all the table rows.

Step 3 You can make changes to an entry field. Position the cursor in the field, and type a new value. Blank out any characters that remain from the previously displayed value.

Step 4 You can make changes to the global options for the job policy. For details, see “Editing or Viewing Job Policy Global Options” on page 4-7.

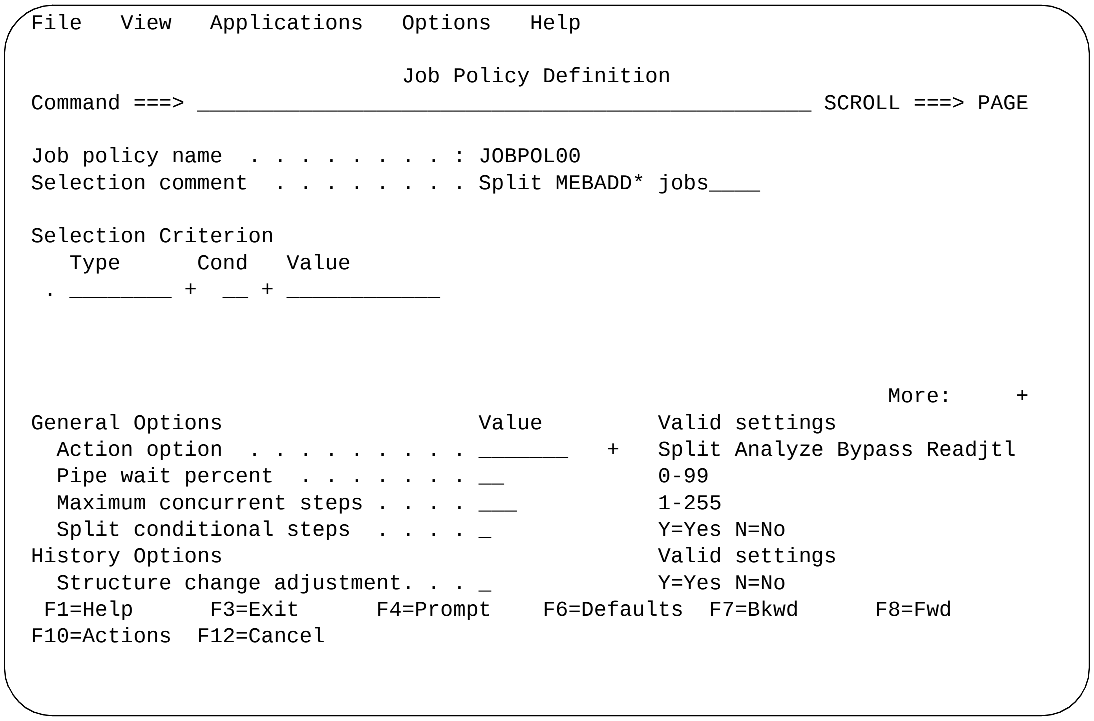
Step 5 You can make changes to the definitions in the job policy. For details, see “Panel Flow for a Job Policy Definition” on page 4-2.

Step 6 You can make changes to the list of definitions by doing the following administrative tasks:

Note: The order of job policy definitions affects Job Optimizer operation.

- To add a new definition to the top of the list, select 1 (New) from the File pull-down, and press **Enter**. The dialog displays the Job Policy Definition panel (Figure 4-4). Edit the new definition.

Figure 4-4 Job Policy Definition Panel



```
File   View   Applications   Options   Help

                             Job Policy Definition
Command ===> ________________________________________________ SCROLL ===> PAGE

Job policy name  . . . . . . . . : JOBPOL00
Selection comment  . . . . . . . . Split MEBADD* jobs____

Selection Criterion
   Type      Cond   Value
 . ________ +  __ + ____________



                                                                   More:     +
General Options                   Value         Valid settings
  Action option  . . . . . . . . _______   +   Split Analyze Bypass Readjtl
  Pipe wait percent  . . . . . . . __            0-99
  Maximum concurrent steps . . . . ___           1-255
  Split conditional steps  . . . . _             Y=Yes N=No
History Options                                 Valid settings
  Structure change adjustment. . . _             Y=Yes N=No
 F1=Help      F3=Exit      F4=Prompt    F6=Defaults  F7=Bkwd      F8=Fwd
F10=Actions  F12=Cancel
```

- To insert (similar to add, but in your choice of list position) a new definition, decide which list definition you want the new definition to be placed after. Position the cursor in the action entry field to the left of your choice of list definition, type **I** (Insert), and press **Enter**. The dialog displays the Job Policy Definition panel (Figure 4-4). Edit the new definition.

- To copy (and paste) a definition, position the cursor in the action entry field to the left of your choice of definition. Type **C** (Copy), and press **Enter**. The dialog response is: Definition copied.

Decide which list definition you want the copied definition to be placed after. Position the cursor in the action entry field to the left of your choice of list definition. Type **P** (Paste), and press **Enter**. The dialog moves the definition from the clipboard to the list.

- To move (cut and paste) a definition, position the cursor in the action entry field to the left of your choice of definition. Type **X** (Cut), and press **Enter**. The dialog response is: Definition cut.

  If the clipboard is empty, the dialog moves the definition from the list to the clipboard.

Decide which list definition you want the cut definition to be placed after. Position the cursor in the action entry field to the left of your choice of list definition. Type **P** (Paste), and press **Enter**. The dialog moves the definition from the clipboard to the list. If you want to edit the moved definition, see "Editing or Viewing a Job Policy Definition" on page 4-9.

- To delete a definition, position the cursor in the action entry field to the left of your choice of definition. Type **D** (Delete), and press **Enter**. The dialog displays a Confirm Delete pop-up, which allows you to confirm the delete request before the definition is removed from the job policy.

Step 7 You can save your changes by using one of two methods:

- Select the File pull-down choice 2 (Save policy). The dialog saves the changed job policy and redisplays the Job Policy Definition panel.

- Press **F3** (Exit). The dialog displays a Confirm Exit pop-up. Type **1** (Save policy changes to disk and exit), and press **Enter**. The dialog saves the changed job policy and displays the MAINVIEW Batch Optimizer Objects List panel with the changed policy at the top of the scrollable area.

Editing or Viewing Job Policy Global Options

You can review or modify the global option values saved for a job policy, including the Job Optimizer response to jobs that match the criteria in one or more definitions, and the options that qualify that response. The global options are explained starting at "Editing or Viewing a Job Policy Definition" on page 4-9.

To edit or view global options for a job policy in the control data set, complete the following steps:

Step 1 Access the Job Policy Definition panel (Figure 4-4 on page 4-6). See "Editing or Viewing a Job Policy" on page 4-4 for information about accessing this panel.

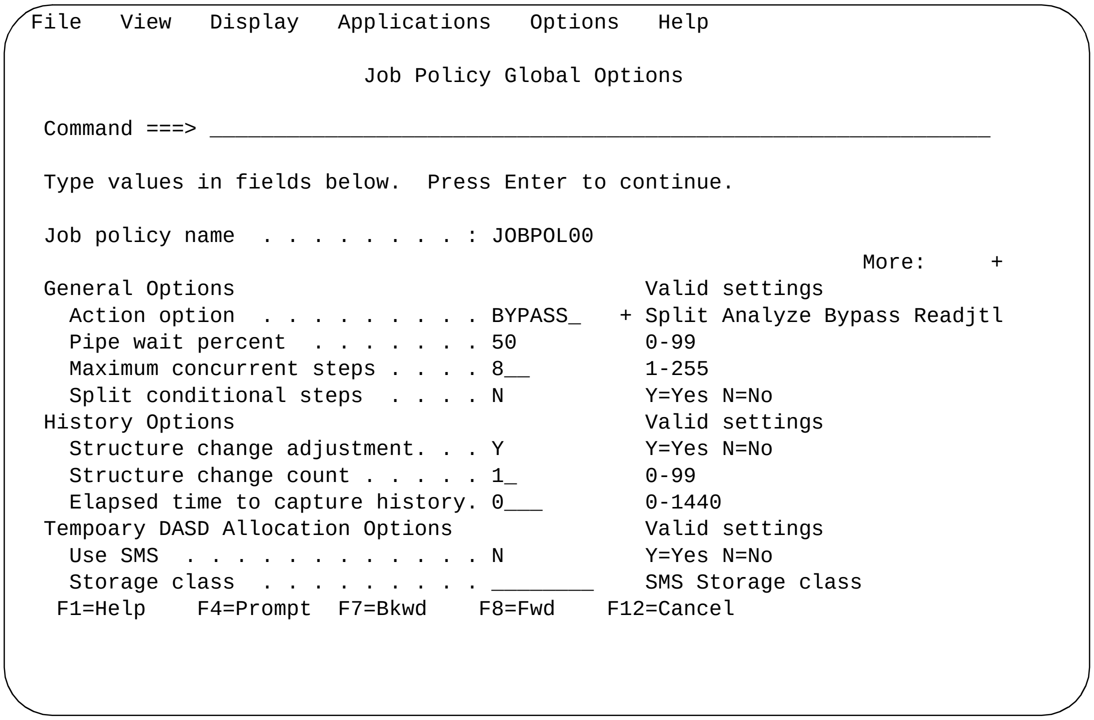
Step 2 Position the cursor on the action bar Options option, and press **Enter**.

The dialog displays the Options pull-down. Type **6** (Job policy global options), and press **Enter**. The dialog displays the Job Policy Global Options pop-up, which allows you to specify or change values that serve as the default field values for each action definition in the job policy and that serve as the action definition for definitions that do not point to a named action definition. The information you can specify or change is the same as on the Job Policy Definition panel (Figure 4-4 on page 4-6).

Figure 4-5 shows an example of the Job Policy Global Options pop-up, where the job policy global options are defined. These sample global options were defined during the customization process.

If you leave performance option fields blank on the Job Policy Definition panel, Job Optimizer uses the values that you specified on the Job Policy Global Options pop-up.

Figure 4-5 Job Policy Global Options Panel



```
File   View   Display   Applications   Options   Help

                        Job Policy Global Options

 Command ===> ____________________________________________________________

 Type values in fields below.  Press Enter to continue.

 Job policy name  . . . . . . . . : JOBPOL00
                                                            More:     +
 General Options                             Valid settings
   Action option  . . . . . . . . . BYPASS_   + Split Analyze Bypass Readjtl
   Pipe wait percent  . . . . . . . 50          0-99
   Maximum concurrent steps . . . . 8__         1-255
   Split conditional steps  . . . . N           Y=Yes N=No
 History Options                             Valid settings
   Structure change adjustment. . . Y           Y=Yes N=No
   Structure change count . . . . . 1_          0-99
   Elapsed time to capture history. 0___        0-1440
 Tempoary DASD Allocation Options            Valid settings
   Use SMS  . . . . . . . . . . . . N           Y=Yes N=No
   Storage class  . . . . . . . . . ________    SMS Storage class
  F1=Help    F4=Prompt  F7=Bkwd    F8=Fwd    F12=Cancel
```

The Job Policy Global Options pop-up is scrollable and may include a More: field.

Step 3 You can make changes to an entry field. Position the cursor in the field, and type a new value. Blank out any characters that remain from the previously displayed value. Repeat this step for each field that you want to edit.

Step 4 Press **Enter** to retain your changes within the dialog, or press **F12** (Cancel) to discard your changes. The dialog displays the Job Policy Definition panel, where you can commit changes to the control data set.

Step 5 You can commit the changes to the control data set by selecting the File pull-down choice 2 (Save Policy). The dialog saves the changed job policy and redisplays the Job Policy Definition panel.

Editing or Viewing a Job Policy Definition

You can review or modify the information saved for a job policy definition, including the Job Optimizer response, to jobs that match the criteria in one or more definitions, and the options that qualify that response. For more information about the available options, see “Job Performance Options” on page 4-20.

To edit or view a definition in a job policy, complete the following steps:

Step 1 Access the Job Optimization Policy panel (Figure 4-3 on page 4-5). See “Editing or Viewing a Job Policy” on page 4-4 for information about accessing this panel.

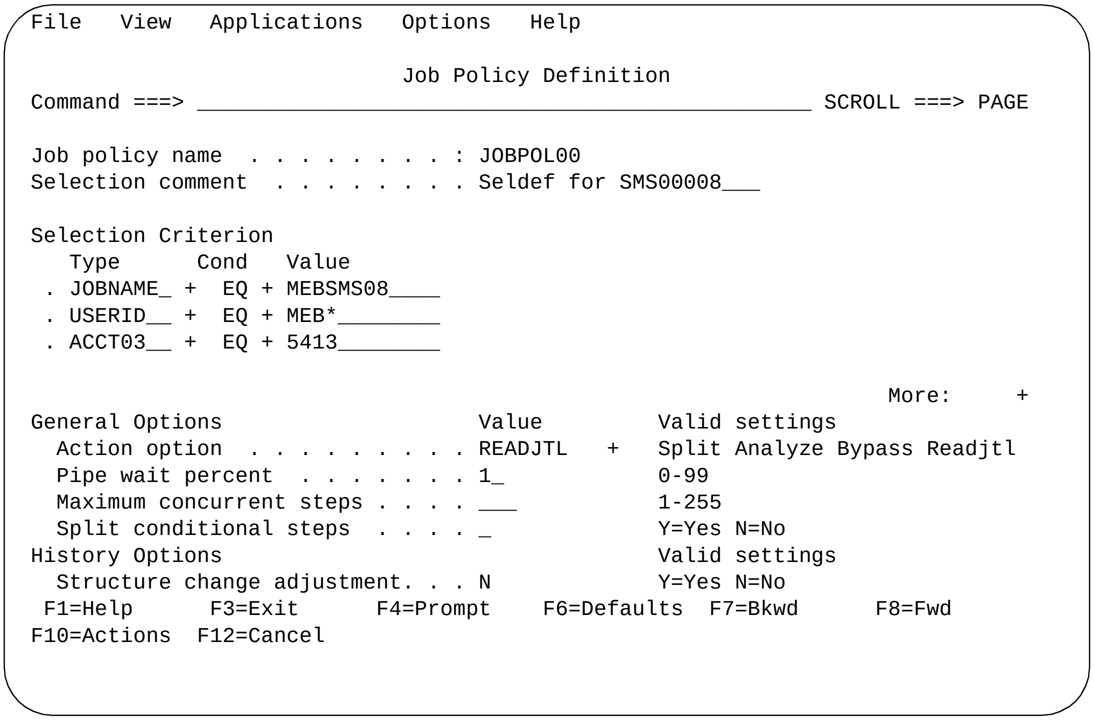
Step 2 Position the cursor in the action entry field to the left of your choice of selection criteria. Type **E** (Edit), and press **Enter**.

The dialog displays the Job Policy Definition panel, which allows you to specify or change action information such as:

- how you want Job Optimizer to respond to jobs that match the criteria in job policy definitions
- where you want Job Optimizer to direct messages for split job steps
- where you want Job Optimizer to allocate temporary and permanent data sets associated with split job steps
- how you want Job Optimizer to handle sysout for split job steps

Figure 4-6 on page 4-10 shows an example of the Job Policy Definition panel, where the definition indicates that Job Optimizer should split job steps for parallel or targeted processing (where possible). This sample definition was defined during the customization process. All other job performance options should come from the job policy global options. See “Editing or Viewing Job Policy Global Options” on page 4-7.

Figure 4-6 Job Policy Definition Panel



```
File   View   Applications   Options   Help

                             Job Policy Definition
Command ===> ________________________________________________ SCROLL ===> PAGE

Job policy name  . . . . . . . . . : JOBPOL00
Selection comment  . . . . . . . . . Seldef for SMS00008___

Selection Criterion
   Type      Cond   Value
 . JOBNAME_ +  EQ + MEBSMS08____
 . USERID__ +  EQ + MEB*________
 . ACCT03__ +  EQ + 5413________

                                                                   More:     +
General Options                       Value         Valid settings
  Action option  . . . . . . . . . . READJTL   +   Split Analyze Bypass Readjtl
  Pipe wait percent  . . . . . . . . 1_            0-99
  Maximum concurrent steps . . . . . ___           1-255
  Split conditional steps  . . . . . _             Y=Yes N=No
History Options                                    Valid settings
  Structure change adjustment. . . . N             Y=Yes N=No
 F1=Help      F3=Exit      F4=Prompt    F6=Defaults  F7=Bkwd      F8=Fwd
F10=Actions  F12=Cancel
```

The Job Policy Definition panel has two scrollable areas and might include a More: field.

Step 3 You can make changes to an entry field. Position the cursor in the field, and type a new value. Blank out any characters that remain from the previously displayed value. Leave blank any field for which you want to use the job policy global option value. Repeat this step for each field you want to edit.

Step 4 You can view a selection list of action options. Position the cursor in the Action option field, and press **F4** (Prompt).

The dialog displays an Action Option List pop-up, which lists actions Job Optimizer can take in response to a batch job. If you want to select an action from the list, position the cursor to the left of your choice of action, and press **Enter**. The dialog displays the Job Policy Definition panel with your choice of action displayed in the Action option field. If you do not want to select an action from the list, press **F12** (Cancel). The dialog redisplays the Job Policy Definition panel.

Step 5 Press **Enter** to retain your changes within the dialog, or press **F12** (Cancel) to discard your changes. The dialog displays the Job Policy Definition panel, where you can commit dialog changes to the control data set.

Step 6 You can commit the changes to the control data set by selecting the File pull-down choice 2 (Save Policy). The dialog saves the changed job policy and redisplays the Job Policy Definition panel.

Editing or Viewing a Job Policy Selection Criteria

You can review or modify the information saved for a job policy definition.

To edit or view a definition in a job policy, complete the following steps:

Step 1 Access the Job Policy Definition panel (Figure 4-6 on page 4-10). See “Editing or Viewing a Job Policy” on page 4-4 for information about accessing this panel.

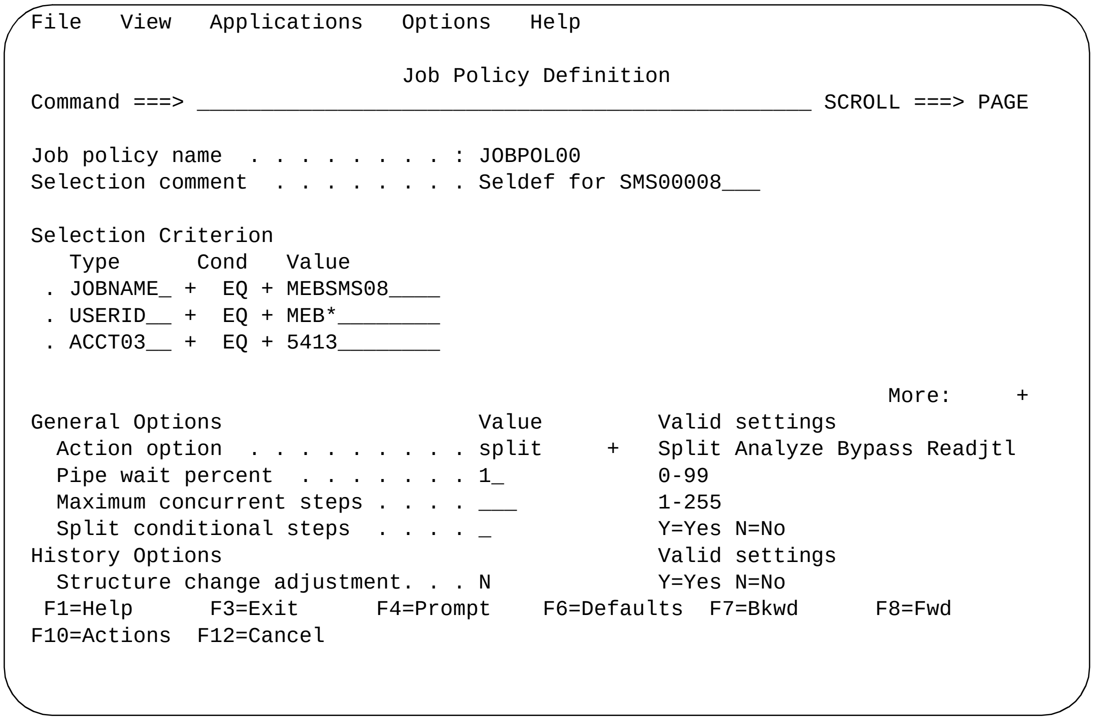
Step 2 Position the cursor in the action entry field to the left of your choice of definition. Type **E** (Edit), and press **Enter**.

The dialog displays the Job Policy Definition panel, which lists the items in a definition. For each item, the list includes the associated type of batch job characteristic, value or value pattern for that characteristic, comparison condition between type and value, and logical connector to the next item. The logical connector is always AND. It is present as a visual cue that all criteria in the definition must be met before the job can be intercepted.

The Job Policy Definition panel also allows you to specify or change selection criterion.

Figure 4-7 shows an example of the Job Policy Definition panel, where three selection items are listed. This sample definition was defined during the customization process. At your site, USERID would be replaced with the TSO/E user ID that was specified by your system programmer. This definition points to an action of SPLIT. Therefore, all batch jobs that are submitted by the specified user and named MEBSM508 will be given job performance improvements (that is, split for parallel or targeted processing, where possible).

Figure 4-7 Job Policy Definition Panel



```
File   View   Applications   Options   Help

                             Job Policy Definition
Command ===> ________________________________________________ SCROLL ===> PAGE

Job policy name  . . . . . . . . . : JOBPOL00
Selection comment  . . . . . . . . . Seldef for SMS00008___

Selection Criterion
   Type      Cond   Value
 . JOBNAME_ +  EQ + MEBSMS08____
 . USERID__ +  EQ + MEB*________
 . ACCT03__ +  EQ + 5413________

                                                                   More:     +
General Options                   Value         Valid settings
  Action option  . . . . . . . . . split     +   Split Analyze Bypass Readjtl
  Pipe wait percent  . . . . . . . 1_            0-99
  Maximum concurrent steps . . . . ___           1-255
  Split conditional steps  . . . . _             Y=Yes N=No
History Options                                 Valid settings
  Structure change adjustment. . . N             Y=Yes N=No
 F1=Help      F3=Exit      F4=Prompt    F6=Defaults  F7=Bkwd      F8=Fwd
F10=Actions  F12=Cancel
```

Step 3 You can make changes to an entry field. Position the cursor in the field, and type a new value. Blank out any characters that remain from the previously displayed value. Repeat this step for each field that you want to edit.

Step 4 You can view a selection list for the action option pointer. Position the cursor in the Action option field, and press **F4** (Prompt).

The dialog displays an Job Policy Action Option pop-up, which lists names of definitions in the job policy. If you want to select a name from the list, position the cursor to the left of your choice of name, and press **Enter**. The dialog displays the Job Policy Definition panel with your choice of name displayed in the action option field. If you do not want to select a name from the list, press **F12** (Cancel). The dialog redisplays the Job Policy Definition panel.

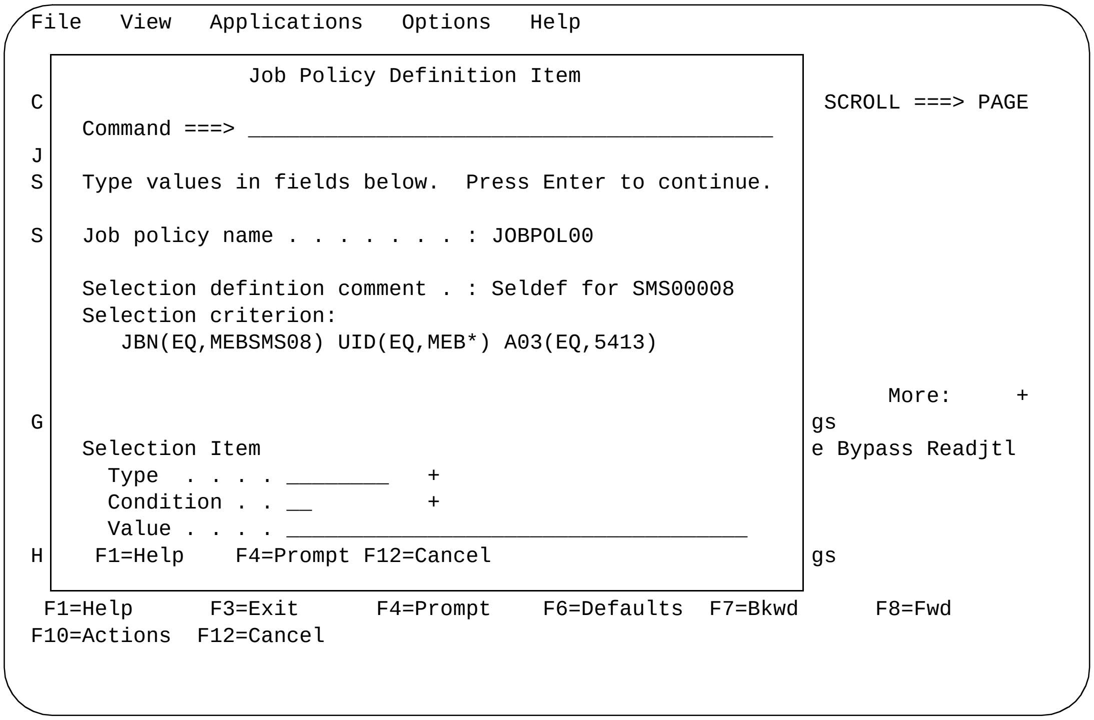
You can type the name of an action option that is not defined, but you must add the action to the job policy before activating a BatchPlex that points to the job policy.

Step 5 You can make changes to the selection criterion. For details, see “Editing or Viewing a Job Policy Selection Item” on page 4-14.

Step 6 You can make changes to the list of selection items by doing the following administrative tasks:

- To add a new item to the top of the list, select 1 (New criteria) from the File pull-down, and press **Enter**. The dialog displays the Job Policy Definition Item pop-up (Figure 4-8). Edit the new item.

Figure 4-8 **Job Policy Definition Item Pop-Up Panel**



```
File   View   Applications   Options   Help
   ┌───────────────────────────────────────────────────────────┐
   │               Job Policy Definition Item                  │
C  │                                                           │ SCROLL ===> PAGE
   │ Command ===> _________________________________________   │
J  │                                                           │
S  │ Type values in fields below.  Press Enter to continue.    │
   │                                                           │
S  │ Job policy name . . . . . . . : JOBPOL00                  │
   │                                                           │
   │ Selection defintion comment . : Seldef for SMS00008       │
   │ Selection criterion:                                      │
   │    JBN(EQ,MEBSMS08) UID(EQ,MEB*) A03(EQ,5413)             │
   │                                                           │
   │                                                           │      More:     +
G  │                                                           │gs
   │ Selection Item                                            │e Bypass Readjtl
   │   Type  . . . . ________   +                              │
   │   Condition . . __         +                              │
   │   Value . . . . ____________________________________      │
H  │  F1=Help    F4=Prompt F12=Cancel                          │gs
   └───────────────────────────────────────────────────────────┘
 F1=Help      F3=Exit      F4=Prompt    F6=Defaults  F7=Bkwd      F8=Fwd
F10=Actions  F12=Cancel
```

- To insert (similar to add, but in your choice of list position) a new item, decide which list item you want the new item to be placed after. Position the cursor in the action entry field to the left of your choice of list item, type **I** (Insert), and press **Enter**. Edit the new item.

Step 7 Press **Enter** to retain your changes within the dialog, or press **F12** (Cancel) to discard your changes. The dialog displays the Job Policy Definition panel, where you can commit dialog changes to the control data set.

Step 8 You can commit the changes to the control data set by selecting the File pull-down choice 2 (Save Policy). The dialog saves the changed job policy and redisplays the Job Policy Definition panel.

Note: Be sure that at least one selection criterion is specified in the job policy. Though it is possible to save an empty job policy in this dialog, you will not be able to activate it.

Editing or Viewing a Job Policy Selection Item

You can review or modify the information saved for a job policy selection criterion, including the item type and value or value pattern.

To edit or view an item in a job policy selection criterion, complete the following steps:

Step 1 Access the Job Policy Definition panel (Figure 4-6 on page 4-10). See "Editing or Viewing a Job Policy Selection Criteria" on page 4-11 for information about accessing this panel.

Notice that the panel displays the selection criterion, which allows you to specify or change the following criteria

- type of batch job characteristic
- comparison condition between type and value
- value or value pattern for the specified batch job characteristic

Note: The logical connector is always AND. It is present as a visual cue that all criteria in the definition must be met before the job can be intercepted.

Step 2 You can make changes to an entry field. Position the cursor in the field, and type a new value. Blank out any characters that remain from the previously displayed value. Repeat this step for each field you want to edit.

For the Value field, you can specify a string that includes a wildcard character asterisk (*) or percent sign (%).

Wildcard characters have the following meaning:

%	A percent sign represents any single character
*	A single asterisk represents zero or more characters
**	A double asterisk represents zero or more characters

When the TYPE field value is JOBCLASS, you cannot specify wildcard characters in the Value field.

Note: For job policies, you can use wildcard characters in MVS Version 5.2 and later.

Step 3 You can view a selection list of item types. Position the cursor in the Type field, and press **F4** (Prompt).

The dialog displays a Selection Type Options pop-up, which lists the characteristics with which you can identify batch jobs. If you want to select an item type from the list, position the cursor to the left of your choice of type, and press **Enter**. The dialog displays the Job Policy Definition Item pop-up with your choice of item type displayed in the Type field. If you do not want to select an item type from the list, press **F12** (Cancel).

Step 4 Press **Enter** to retain your changes within the dialog, or press **F12** (Cancel) to discard your changes. The dialog displays the Job Policy Definition panel, where you can commit dialog changes to the control data set.

Step 5 You can commit the changes to the control data set by selecting the File pull-down choice 2 (Save Policy). The dialog saves the changed job policy and redisplays the Job Policy Definition panel.

Activating Job Policy Definitions

When you use the MAINVIEW Batch Optimizer dialog to edit a job policy, you must activate the policy to have your changes take effect. Saving an edited policy does not cause the changes to take effect.

Activating a Job Policy

Your site’s initial job policy was created during the MAINVIEW Batch Optimizer customization process. The job policy becomes active when you start the MAINVIEW Batch Optimizer subsystem on the images in the BatchPlex. The name of the job policy is defined within the BatchPlex definition.

For a new job policy, you must activate the policy on each MVS image in the BatchPlex.

You can use a MAINVIEW Batch Optimizer command or the MAINVIEW Batch Optimizer dialog to activate a job policy. Using the commands or dialog, however, activates the policy for the current activation of the BatchPlex only. When you restart the MAINVIEW Batch Optimizer subsystem the policy is identified in the BatchPlex definition become active again.

To make a job policy permanent, you must change the job policy in the BatchPlex definition.

BMC Software, Inc., Confidential and Proprietary Information

When you activate a job policy using a command or the dialog, the MAINVIEW Batch Optimizer subsystem responds with messages such as the following to indicate whether the activation was successful

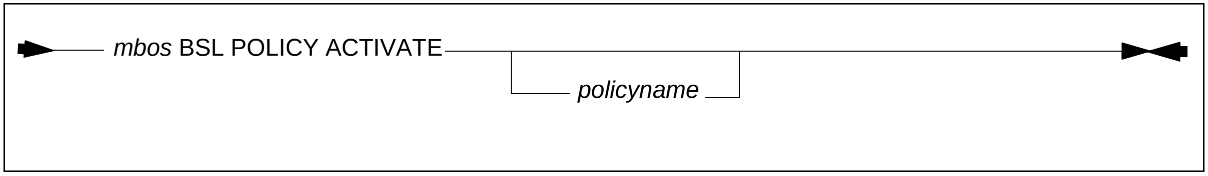
- policy was activated successfully
- policy was not found in the control data set
- policy contains an error

Activating a Job Policy Using a Command

To activate a job policy using a command, enter the BSL POLICY ACTIVATE command from the MVS console.

The syntax of this command is:

Figure 4-9 BSL Policy Activate



where *mbos* is the ID of the MAINVIEW Batch Optimizer subsystem that is active on the MVS image, and policyname is the name of an existing job policy definition in the control data set.

Activating a Job Policy Using the Dialog

To activate a job policy definition using the dialog, complete the following steps:

Step 1 Access the MAINVIEW Batch Optimizer Objects List panel Figure 4-2 on page 4-4. For information on editing or viewing a BatchPlex definition, see the *MAINVIEW Batch Optimizer Installation Manual*.

Step 2 Position the cursor in the action entry field to the left of your choice of job policy. Type **A** (Activate), and press **Enter**.

The dialog displays the Policy Activation BatchPlex Name pop-up.

Step 3 Type the name of a BatchPlex on which to activate the policy or definition, and press **Enter**.

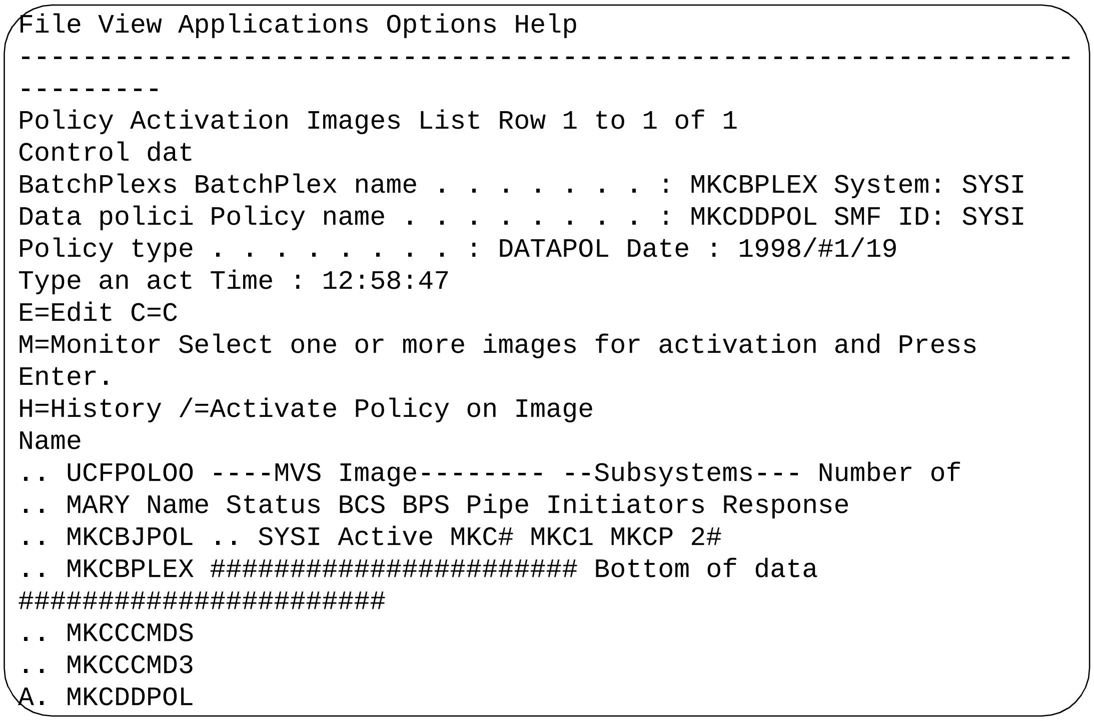
You can view a selection list of item types. Position the cursor in the BatchPlex name field, and press **F4** (Prompt).

The dialog displays a BatchPlex Definition List pop-up, which lists the available BatchPlexes. To select an item from the list, position the cursor to the left of a name, and press **Enter**. The dialog displays the Policy Activation BatchPlex Name pop-up with your choice of BatchPlex displayed in the BatchPlex name field. If you do not want to select an item type from the list, press **F12** (Cancel). The dialog redisplays the Policy Activation BatchPlex Name pop-up.

Note: The BatchPlex you specify must be active on the same MVS image as your dialog session and must be defined in the control data set allocated to your dialog session.

The dialog displays the Policy Activation Images List pop-up (Figure 4-10).

Figure 4-10 Policy Activation Images List Pop-Up



```
File View Applications Options Help
------------------------------------------------------------------
---------
Policy Activation Images List Row 1 to 1 of 1
Control dat
BatchPlexs BatchPlex name . . . . . . . : MKCBPLEX System: SYSI
Data polici Policy name . . . . . . . . : MKCDDPOL SMF ID: SYSI
Policy type . . . . . . . . : DATAPOL Date : 1998/#1/19
Type an act Time : 12:58:47
E=Edit C=C
M=Monitor Select one or more images for activation and Press
Enter.
H=History /=Activate Policy on Image
Name
.. UCFPOLOO ----MVS Image-------- --Subsystems--- Number of
.. MARY Name Status BCS BPS Pipe Initiators Response
.. MKCBJPOL .. SYSI Active MKC# MKC1 MKCP 2#
.. MKCBPLEX ###################### Bottom of data
#######################
.. MKCCCMDS
.. MKCCCMD3
A. MKCDDPOL
```

Step 4 Position the cursor in the action entry field to the left of each MVS Image in the BatchPlex on which to activate the policy or definition. Type / (Activate Policy on Image) next to each applicable MVS Image, and press **Enter**.

The Response field for each of the selected MVS images indicates if the policy or definition was successfully activated.

Step 5 Press **F12** (Cancel) or **F3** (Exit) to return to the MAINVIEW Batch Optimizer Objects List panel.

Determining Which Policy Is Active

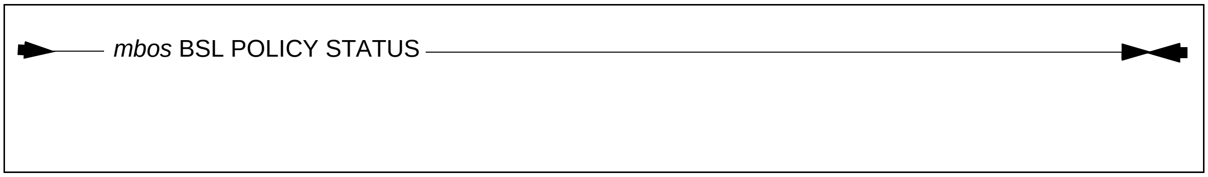
Because a new job policy or UCF definition can be activated at any time, the policy or UCF definition that is identified in the BatchPlex definition might not be the one that is currently in effect. Therefore, before making changes to a policy or UCF definition based on current activities at your site, you should determine which definitions are active.

Determining the Active Job Policy

To determine which job policy is active, enter the BSL POLICY STATUS command from the MVS console.

The syntax of this command is:

Figure 4-11 BSL Policy Status



mbos BSL POLICY STATUS

mbos is the ID of the MAINVIEW Batch Optimizer subsystem that is active on the MVS image.

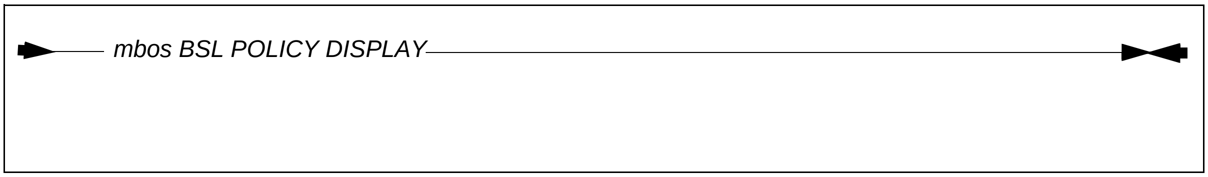
The MAINVIEW Batch Optimizer subsystem responds with messages that indicate the name of the active job policy. Also, the active BatchPlex definition is displayed.

Determining Where the Definition for a Job Is Written

When a job terminates, the job policy definition associated with the job is written. The BSL POLICY DISPLAY command allows you to specify where Job Optimizer is to write the job policy definition that is associated with a job. An example of the command syntax appears on page 4-19.

The syntax of this command is:

Figure 4-12 BSL Policy Display



```
mbos BSL POLICY DISPLAY
```

mbos is the ID of the MAINVIEW Batch Optimizer subsystem that is active on the MVS image, and the option is one of the following:

SYSMSG The job policy definition is written to the job message queue

JOBLOG The job policy definition is written to the joblog.

BOTH The job policy definition is written to both the joblog and the job message queue.

NONE The job policy definition is not written.

STATUS This option displays the current setting of this command.

The default value is SYSMSG.

Job Performance Options

Job Optimizer provides job performance options to allow you to control how job performance processing occurs. The options are located in one of the following locations:

- Job Policy Definition panel, which you use to define job policy definition statements

- Job Policy Global Options pop-up, which you use to define the default (global) value for each option

This section describes the purpose of each job performance option. On the pop-up and panel, the options are divided into groups. The following sections discuss the options in each group:

- general options
- history options
- temporary DASD allocation options
- permanent DASD allocation options
- sysout handling options
- split step message options
- step termination options

General Options

The General options determine the following:

- the type of job performance processing to accomplish
- the maximum amount of time to allow for the creation of a pipe
- how to respond to conditional JCL parameters

Action Option

The Action option has the most influence on job performance processing; this option determines the type of job performance processing that Job Optimizer is to provide for selected batch jobs that are initiated in a BatchPlex.

With this option, you can specify how you want Job Optimizer to respond to jobs that match the selection criteria.

Values

The action option has the following values:

BYPASS

Do not apply performance processing to this job. BMC Software recommends that you specify the BYPASS action for jobs that are not good candidates for job performance processing.

SPLIT

Apply performance processing to this job.

ANALYZE

Evaluate this job as a candidate for future Job optimization processing. If a job is a good candidate for job performance processing, choose the ANALYZE action for the job, to collect its history information. Later, reset the action option for the job to SPLIT, to obtain the benefits of job performance processing.

READJTL

Apply user-specified JTL statements to this job. The READJTL action causes Job Optimizer to read JTL statements that you place in the job stream. JTL statements are specialized Job Optimizer control cards in the form of JCL comment statements. For more information about JTL, see “Understanding Job Policy Definitions” on page 4-2. With JTL statements, you can control performance processing at the job, job step, or DD statement levels, as follows:

- At the job level, you can direct Job Optimizer to bypass, analyze, or split batch jobs.
- At the step level, you can direct Job Optimizer to perform the following:
- Bypass the step
- Split the step regardless of whether it contains data constraints that would normally prevent Job Optimizer from splitting it
- Target the step to a specific MVS image
- Not run additional steps until the step completes
- Allow the step to run in parallel with previous steps.
- At the DD level, you can assign a specific access intent (input, output, or update) to a DD statement. Doing so helps Job Optimizer to relieve constraints that might prevent batch jobs from splitting.

Default

Analyze

Policy Keyword

ACTION={SPLIT | READJTL | ANALYZE | BYPASS}

Tape Splitting Option

The Tape Splitting option determines whether Job Optimizer should split job steps that have tape data sets.

Values

This option has the following possible values:

Y

Job Optimizer should split job steps that have tape data sets whenever possible.

N Job Optimizer should not split job steps that have tape data sets.

Default N

Policy Keyword TAPESPLIT={Y | N}

Pipe Wait Percent Option

The Pipe Wait Percent option allows you to specify the maximum amount of time that Job Optimizer allows for pipe creation for the transfer of data from a writer step to a reader step. If a pipe is not created within the specified amount of time, Job Optimizer will use the original media to complete the data transfer.

With this option, you can specify how long Job Optimizer should wait for successful introduction of a data pipe for transfer of data from a writer step to a reader step.

Values The pipe wait percent option has the following possible numeric values:

1-99 Job Optimizer waits a minimum elapsed time equal to the product of this value and the average elapsed time (from history data set records) for the writer step.

0 Job Optimizer waits indefinitely.

Default 2

Policy Keyword WAITCNT={0–99}

Maximum Concurrent Steps Option

The Maximum Concurrent Steps option lets you to specify the maximum number of steps per job that Job Optimizer will execute in parallel.

Values This option has the following possible numeric values:

1–255 The number of steps per job that Job Optimizer will execute in parallel will not exceed the value specified in this field.

Default 8

Policy Keyword MAX_SPLIT={1–255}

Split Conditional Steps Option

The Split Steps With Condition Codes option allows you to specify how Job Optimizer responds when it encounters conditional processing such as COND= or IF-then-ELSE JCL statements.

Many conditional parameters are included in jobs to account for “normal” processing. For example, a program sets a condition code that affects future steps. Other conditional are included in jobs to account for “abnormal” processing. For example, a program requires a sorted file, but the sort fails.

With this option, you can specify how Job Optimizer responds when it encounters conditional processing.

Values This option has the following possible values:

N Job Optimizer honors the condition code associated with a job step. For a job-level condition code, the job step is not split. For a step-level condition code, the job step is split once the named job step completes. If the COND= parameter does not name a job step, all previous job steps must complete before the job step is processing, you would use this value to specify that Job Optimizer honor the COND= parameter.

Y Job Optimizer ignores the condition code constraint for a job step. Job Optimizer, therefore, splits the step if there are no other constraints for the job step.

Default N

Policy Keyword CONDCODES={Y|N}

Transport Mechanism Option

The Transport Mechanism option determines how Job Optimizer will resolve data dependencies when splitting job steps.

Values This option has the following possible values:

AUTO Job Optimizer chooses Job Optimizer Pipes or Shared Record Positioning (SRP)

SRP Job Optimizer uses SRP to resolve data dependencies when splitting job steps.

JOP Job Optimizer uses Job Optimizer Pipes to create data pipes between steps with data dependencies.

Default AUTO

Policy Keyword TRANSPORT={SRP|JOP|AUT}

History Options

The History options control when Job Optimizer begins splitting job steps that have constraints by relieving those constraints based on history patterns.

Job Optimizer starts a stability counter at zero when analyzing a job for the first time. It increments the counter each time it analyzes the job again and the patterns match. The history count and adjustment options determine:

- the value that the stability counter must reach before the constraints can be relieved
- the circumstances that cause the stability counter to be reset to zero

Structure Change Adjustment Option

Job Optimizer starts data access counters at zero when analyzing a job for the first time. Job Optimizer increments or resets the counters each time it analyzes the job again. The History Options control allows you to specify whether Job Optimizer should reset the stability counter to zero when it encounters job structure changes (changes in the pattern of EXEC and DD statements).

Values This option has two values:

Y Job Optimizer resets the stability counter to zero after a job structure change.

N Job Optimizer leaves the stability counter at its current value after a job structure change.

Default Y

Policy Keyword STRCHG={Y|N}

Structure Change Count

With the Structure Change Count option, you set the value of the stability counter. The stability counter indicates the number of times the job structure and data access patterns must match, or remain static, before data constraints can be relieved based on those patterns.

Values This option has two values:

1–99 When the stability counter value reaches this number for a job, Job Optimizer begins relieving the data constraints and scheduling the job steps for appropriate parallel or targeted execution.

0 Data constraints cannot be relieved. For job steps that have data constraints, Job Optimizer will not schedule the steps together.

Default 1

Policy Keyword FACTOR1={0–99}

Elapsed Time To Capture History Option

With the Elapsed Time To Capture History option, you can limit activity in the history data set. Use this option to specify the amount of time (in minutes) that must pass before Job Optimizer starts collecting history data for a job. Once a job meets this threshold, the elapsed time value is no longer used as a criterion.

Values This option accepts the following values:

0 Record data to the history data set as soon as the job starts.

1–1440

Do not start recording history unless the job reaches or exceeds the specified elapsed time.

Default 0

Policy Keyword HISTCAPTURE={0–1440}

Temporary DASD Allocation Options

Five options enable you to identify where you want Job Optimizer to allocate spill data sets associated with system output for split job steps.

Use SMS Option

For temporary allocations of spill data sets, you can indicate whether to use the SMS options or the unit name option for data set allocation

Values This option has two values:

Y Job Optimizer should use SMS options to allocate temporary spill data sets. When using this option, you must complete the relevant SMS fields either on this panel or via the global defaults.

N Job Optimizer should use the esoteric unit name specified in the unit name field. When using this option, you must provide a value in the unit name field either on this panel or via the global default.

Default N

Policy Keyword USESMS={Y|N}

Storage Class Option

For temporary allocations of spill data sets, you can specify an SMS storage class for spill data sets that Job Optimizer allocates. The SMS storage class value should follow JCL naming conventions.

Default none

Policy Keyword STORCLAS=

Data Class Option

For temporary allocations of spill data sets, you can specify an SMS data class for spill data sets that Job Optimizer allocates. The SMS data class value should follow JCL naming conventions.

Default none

Policy Keyword DATACLAS=

Management Class Option

For temporary allocations of spill data sets, you can specify an SMS management class for spill data sets that Job Optimizer allocates. The SMS management class value should follow JCL naming conventions.

Default	none
Policy Keyword	MGMTCLAS=

Unit Name Option

For temporary allocations of spill data sets, you can specify an esoteric unit name for spill data sets that Job Optimizer allocates. The work unit name value should follow JCL naming conventions.

Default	SYSDA
Policy Keyword	WORKUNIT=

SYSOUT Handling Options

The SYSOUT handling options allow you to indicate when Job Optimizer should merge system output and identify where you want Job Optimizer to retain system output for split job steps before spooling.

With the options in the sysout handling options group, you can specify the following:

- how to arrange system output
- the number of lines of sysout data to retain in memory before spilling
- the number of lines of sysout data that each spill data set is to hold
- the amount of sysout data to generate

Spool Scheduling Option

With the Spool Scheduling option, you can specify the manner in which Job Optimizer arranges system output from split steps into a single job output. You can specify when Job Optimizer should merge system output from split steps into a single job output.

Values

This option has two possible values:

ASAP Job Optimizer merges system output for a job in the order of step completion. You may want to specify this value to have optimized performance. However, ASAP spooling may result in a different order of step output from one job run to the next.

DEFER Job Optimizer merges system output for a job in the same order as for normal sequential processing. You may want to specify this value to have a predictable order of job step output. However, defer spooling requires additional DASD and CPU resources.

Default

DEFER

Policy Keyword

WHENSPOOL={ASAP|DEFER}

Segment Spill Point Option

With the Segment Spill Point option, you can specify the number of lines of sysout data that Job Optimizer is to retain in memory before spilling to a spill data set. The value specified is that many thousands of lines of sysout data. For example, a value of 1 is 1000 lines and a value of 5 is 5000 lines. BMC Software recommends a zero value if you specify defer spooling.

Values

You may specify the following values for this field.

0–999 Job Optimizer will retain this many thousand lines of system output in memory before spilling to a spill data set.

Default

10

Policy Keyword

SEGSPILL={0–999}

Lines Per Spill Data Set Option

With the Lines Per Spill Data Set option, you can specify the number of lines of sysout data that each spill data set is to hold. Specify a numeric value that represents the number of 133-byte records in increments of 1000.

Values You may specify the following values for this field.

0–999 Each spill data set is to hold this many thousand lines of 133-byte records.

Default 100

Policy Keyword DSSSPILL={0–999}

Maximum Sysout Lines Per File Option

With the Maximum Sysout Lines Per File option, you can limit the amount of sysout data generated by a DD statement. You can use this option to prevent a job from exhausting auxiliary storage.

Specify a numeric value from 1–999. This value is multiplied by 1000 and then multiplied by 133 (which is the number of bytes per record) to establish the number of bytes used to support the data stream for the sysout. If a job exceeds this limit, the job ends with a U2400 Abend.

For example, choosing a maximum sysout lines per file value of 10 restricts Job Optimizer to generating no more than 1330000 bytes of sysout data for any DD statement in a job. Since the records are 133 bytes each, at least 10000 sysout records can be generated. Because of Job Optimizer's disk caching and in-memory compression, however, many more records can potentially be generated.

Values This option has the following possible numeric values:

0–999 A DD statement can generate this many thousand lines of sysout data before the job terminates with a 222 Abend.

Default 50

Policy Keyword MAXSYSOUT={0–999}

Split Step Message Options

With these options, you can specify where Job Optimizer is to write messages for split job steps and when it should generate a step summary report. You can direct termination messages, joblog messages, and system log messages through the use of these options.

BMC Software recommends that you direct messages to the job message queue to reduce console message traffic. Until you are familiar with the new messages, however, you may want to direct them to both locations.

Termination Option

With the Termination option, you can specify were Job Optimizer is to write termination messages.

Values This option has the following possible values:

JOBLOG Job Optimizer will write termination messages to the joblog.

SYSMSG Job Optimizer will write termination messages to the job message queue.

BOTH Job Optimizer will write termination messages to both the joblog and the job message queue.

NONE Job Optimizer will not write any messages.

Default SYSMSG

Policy Keyword TERMMSG={JOBLOG|SYSMSG|BOTH|NONE}

Split Joblog Option

The Split Joblog option lets you specify where Job Optimizer is to write joblog messages.

Values This option has the following possible values:

JOBLOG Job Optimizer will write joblog messages to the joblog.

SYSMSG Job Optimizer will write joblog messages to the job message queue.

BOTH Job Optimizer will write joblog messages to both the joblog and the job message queue.

NONE Job Optimizer will not write any joblog messages.

Default JOBLOG

Policy Keyword SPJOBLOG={JOBLOG | SYSMSG | BOTH | NONE}

Split Sysmsg Option

The Split Sysmsg option lets you specify where Job Optimizer is to write system log messages.

Values This option has the following possible values:

JOBLOG Job Optimizer will write system log messages to the system log.

SYSMSG Job Optimizer will write system log messages to the job message queue.

BOTH Job Optimizer will write system log messages to both the system log and the job message queue.

NONE Job Optimizer will not write any system log messages.

Default SYSMSG

Policy Keyword SPSYSMSG={JOBLOG | SYSMSG | BOTH | NONE}

Step Summary Option

The Step Summary option lets you specify when Job Optimizer includes summary messages with the termination messages.

Values

This option has the following possible values:

ALWAYS Job Optimizer will include summary messages with termination messages under all circumstances.

ERROR Job Optimizer will include summary messages with termination messages only when an error occurs.

NEVER Job Optimizer will never include summary messages with termination messages.

Default

ALWAYS

Policy Keyword

STEPSUMM={ALWAYS | ERROR | NEVER}

Step Termination Options

These options control how Job Optimizer processes step termination for steps that execute on a Cross-System Image manager (XIM) initiator. They also allow you to indicate how Job Optimizer terminates split job steps that logically appear before or after a step that abends or fails due to a JCL error.

You can instruct Job Optimizer to terminate split steps in normal sequential job order or as they complete. You can also instruct Job Optimizer to fail job steps that are sequentially before or after a failing job step in sequential order.

Sequential Step Termination Option

The Sequential Step Termination option lets you specify whether Job Optimizer should ensure that split job steps terminate in normal sequential order.

Values Three values are possible:

Y Job Optimizer leaves a split step in the job step initiator until the shadow step executes. With this value, the results of the job when a step fails will more closely match normal sequential processing. However, this approach may require more job step initiators.

N Job Optimizer frees the job step initiator after program execution has finished for a split step. With this value, less job step initiators may be required. However when a job step fails, a step that occurs later in the job JCL may have already completed execution and terminated. Unpredictable job results can include data sets that are cataloged or updated differently than with normal sequential processing.

O Job Optimizer frees the job step initiator after program execution has finished for a split step. However, if a step fails, Job Optimizer will invoke conditional data set dispositions for future steps that have already completed.

Default O

Policy Keyword SYNFUTCMP={Y | N | O}

Fail Future Steps On Abend Option

The Fail Future Steps On Abend option lets you specify whether Job Optimizer should terminate (with Abend code 222) job steps that are executing at the same time as a step that abends if the steps appear sequentially *after* the abending step.

Values There are two possible values:

Y Job Optimizer fails executing “future” steps.

N Job Optimizer allows executing “future” steps to continue executing.

Default Y

Policy Keyword ABNFUTCMP={Y | N}

Fail Past Steps On Abend Option

The Fail Past Steps On Abend option lets you specify whether Job Optimizer should terminate (with Abend code 222) job steps that are executing at the same time as a step that Abends if the steps appear sequentially before the Abending step.

Values There are two possible values:

Y Job Optimizer fails executing “past” steps.

N Job Optimizer allows executing “past” steps to continue executing.

Default N

Policy Keyword ABNPASCMP={Y | N}

Fail Future Steps On JCL Error Option

The Fail Future Steps On JCL Error option lets you specify whether Job Optimizer should terminate (with Abend code 222) job steps that are executing at the same time as a step that fails with a JCL error if the steps appear sequentially after the failing step.

Values There are two possible values:

Y Job Optimizer fails executing “future” steps.

N Job Optimizer allows executing “future” steps to continue executing.

Default Y

Policy Keyword JCLFUTCMP={Y | N}

Fail Past Steps On JCL Error Option

The Fail Past Steps On JCL Error option lets you specify whether Job Optimizer should terminate (with Abend code 222) job steps that are executing at the same time as a step that fails with a JCL error if the steps appear sequentially before the failing step.

Values There are two possible values:

Y	Job Optimizer fails executing “past” steps.
N	Job Optimizer allows executing “past” steps to continue executing.

Default N

Policy Keyword JCLPASCMP={Y|N}

BMC Software, Inc., Confidential and Proprietary Information

Chapter 5 Defining the User Control Facility

This chapter describes how to view or modify User Control Facility (UCF) definitions online. Included is information about the panels and dialog actions that you use to accomplish these tasks. This chapter discusses the following topics:

BMC Software, Inc., Confidential and Proprietary Information

Using the User Control Facility

The User Control Facility (UCF) provides you with an additional means of controlling the Job Optimizer component of MAINVIEW® Batch Optimizer. UCF is a set of keywords that you specify in a member of the control data set. UCF applies to batch jobs only; it does not affect MAINVIEW Batch Optimizer's performance processing for data set I/O requests.

About UCF Members

In a UCF member, the entries you specify modify Job Optimizer's processing of particular batch jobs, DDnames, or allocation esoterics, in this order.

Your entries can accomplish the following tasks:

- override Job Optimizer's default processing
- perform event processing
- change VIO from one name to another
- identify esoteric disk or tape

You can create multiple UCF members within a control data set; however, you can have only one active UCF member in a BatchPlex at any time.

While a UCF definition provides a default level of processing options for Job Optimizer, you can override UCF definitions, in turn, with JTL statements (see Chapter 7, "Job Transformation Language"). Also, you cannot use UCF definitions to override existing splitting constraints or data constraints.

Overriding Default Processing

To override the default processing of a job step, you must specify a program name or ddname for Job Optimizer to look for. You must also specify an action for Job Optimizer to take when a specified name is detected. Begin each entry with a PROGRAM=pgmname, DDNAME=ddname, or EVENT=CHANGE statement, where pgmname is a program to be screened and ddname= is a ddname to be recognized.

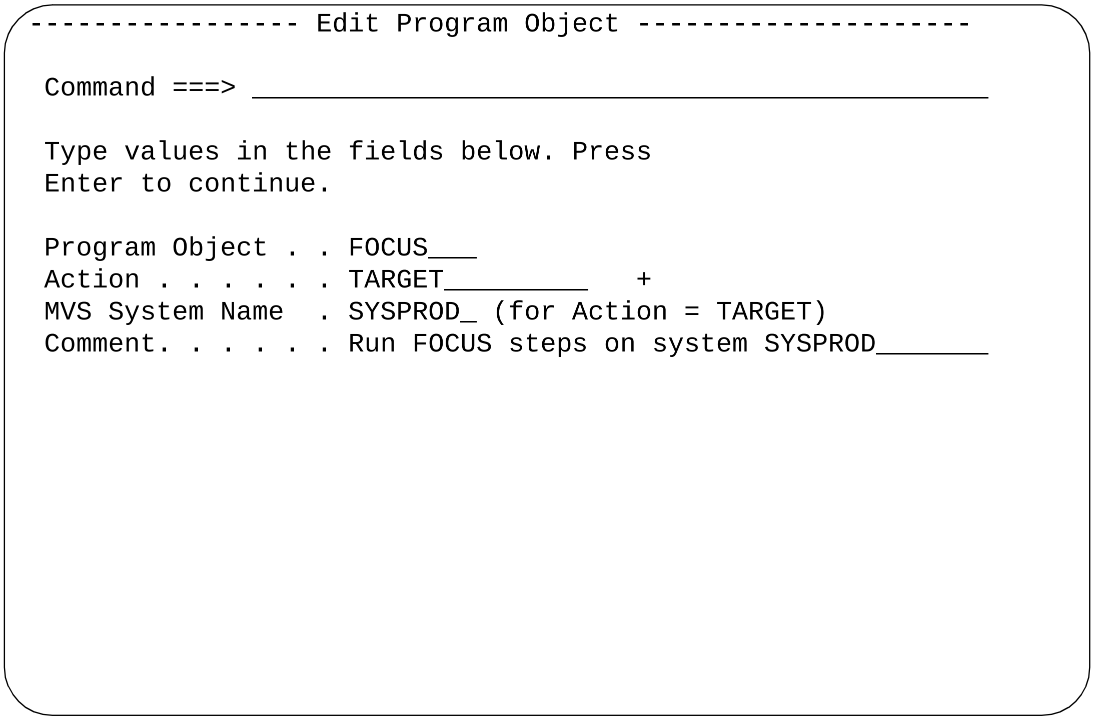
BMC Software, Inc., Confidential and Proprietary Information

Performing Event Processing

For each EVENT statement, specify information about the allocation esoteric.

For example, to always target a program called FOCUS to its originating MVS image, you can add the entry shown in Figure 5-1.

Figure 5-1 Edit Program Object Panel



```
----------------- Edit Program Object ---------------------

 Command ===> ______________________________________________

 Type values in the fields below. Press
 Enter to continue.

 Program Object . . FOCUS___
 Action . . . . . . TARGET_________   +
 MVS System Name  . SYSPROD_ (for Action = TARGET)
 Comment. . . . . . Run FOCUS steps on system SYSPROD_______
```

Changing VIO Devices to non-VIO Devices

If a step allocates a data set on a VIO device, Job Optimizer cannot split the step. You can change the allocation to a non-VIO device by using the EVENT=CHANGE statement in a UCF member.

For example, if you know a data set uses a VIO device named VIO, you can change it to a non-VIO device named SYSDA using the following statement:

```
EVENT=CHANGE
EVENTID=UNIT
FROM=VIO
TO=SYSDA
```

As another example, if your storage group ACS routine assigns a storage group of SMSVIO, you could have Job Optimizer override those assignments with another storage group. This allows your ACS routines to remain coded as they are, and only affects those jobs that are under the control of Job Optimizer. You would include the following statements in your UCF member:

```
EVENT=CHANGE
EVENTID=STORGRP
FROM=SMSVIO
TO=SMSNOVIO
```

Identify Esoteric as Disk or Tape

Job Optimizer needs to know if an esoteric unit name is disk or tape. If your site uses products that allow dynamic definition of esoteric names, it may be necessary to add UCF entries so that Job Optimizer recognizes the unit name. For more information, see “User Control Facility” on page 2-5.

About the Internal UCF Member

During initialization, MAINVIEW Batch Optimizer reads an internal UCF table to set options for certain programs. Then, MAINVIEW Batch Optimizer activates your site's default UCF member, checking it for additions to, or overrides of, the internal UCF table. (See “About UCF Members” on page 5-2 for the content of the internal UCF table).

Your site's UCF definitions complement these internal UCF definitions. If your site defines a UCF entry for a program that is already defined in the internal list, your site's UCF entry overrides the internal entry.

Changing and Creating UCF Members

When initialization is complete, you can change the active UCF member through the BSL UCF ACTIVATE command. When you reinitialize MAINVIEW Batch Optimizer, however, the system reverts to using the previously active UCF member. (For more information about activating the UCF definition, “Activating a UCF Definition Using a Command” on page 5-9.)

During job analysis, Job Optimizer scans the active UCF member to determine whether any site-specific actions are to be taken for a given job step or DD statement. During job processing, Job Optimizer scans the active UCF member for overriding instructions for the program and DDnames that it encounters while processing JCL statements.

You can create a UCF definition to bypass programs that are unsuited for batch job performance processing. Rather than specifying selection criteria for a job in all job policies that might be active on an image, you can bypass the job at the installation level by using a UCF definition.

When a batch job is initiated in a BatchPlex, MAINVIEW Batch Optimizer checks the selection criteria you specify in the UCF definition before reading the job policy. When MAINVIEW Batch Optimizer encounters selected installation conditions (program names, DDnames, or allocation esoterics), MAINVIEW Batch Optimizer performs an action that you specify in a UCF definition (BYPASS, TARGET, or NOOP).

As a result, a UCF definition allows you to specify additional information for a specific job. Or, you can specify the NOOP action to instruct MAINVIEW Batch Optimizer to use the job policy’s action definitions.

For example, a program that uses control cards to direct its behavior (such as SAS or IDCAMS) may not be a suitable candidate for MAINVIEW Batch Optimizer performance processing (MAINVIEW Batch Optimizer cannot determine how the control cards are to be used and cannot split or target the job steps).

For information about creating UCF definitions, see sample member UCFPOL00 in data set SAMPLIB.

Registering User Control Facility Definitions

A UCF definition specifies the actions to occur when a job encounters selected installation conditions (programs, data sets, device types). The UCF definition complements the definitions that you specify in the job policy definition.

During MAINVIEW Batch Optimizer customization, a sample UCF definition was created in the control data set.

This section describes dialog panels and how you can use them to review or modify a UCF definition in a control data set.

BMC Software, Inc., Confidential and Proprietary Information

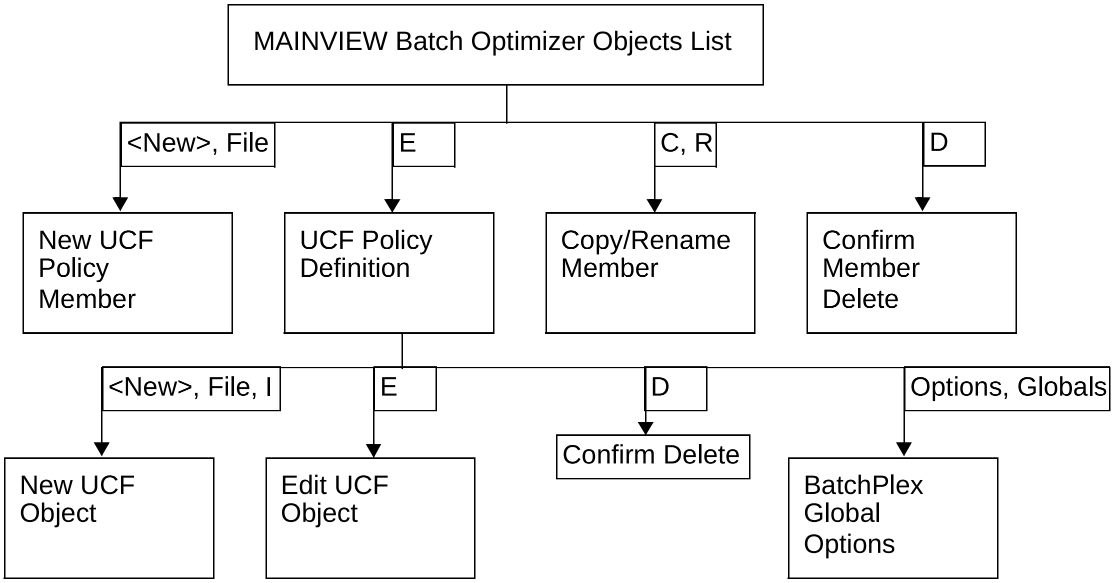
Additional details are available in the following locations:

- prolog of member UCFPOL00 of data set SAMPLIB. This self-documenting member describes the coding requirements of a UCF definition
- online help facility

Panel Flow for a UCF Definition Registration

Figure 5-2 shows the panel flow in the UCF definition registration area of the MAINVIEW Batch Optimizer dialog. The figure is the basic panel flow; it indicates the action code you type (**E**, **I**, **C**, **R**, **D**) or the pull-down option (File) you select to move from one panel or pop-up to another.

Figure 5-2 UCF Definition—Dialog Panel Flow



Editing or Viewing a UCF Definition

You can review or modify the information saved for a UCF definition.

To edit or view information for a UCF definition in the control data set, complete the following steps:

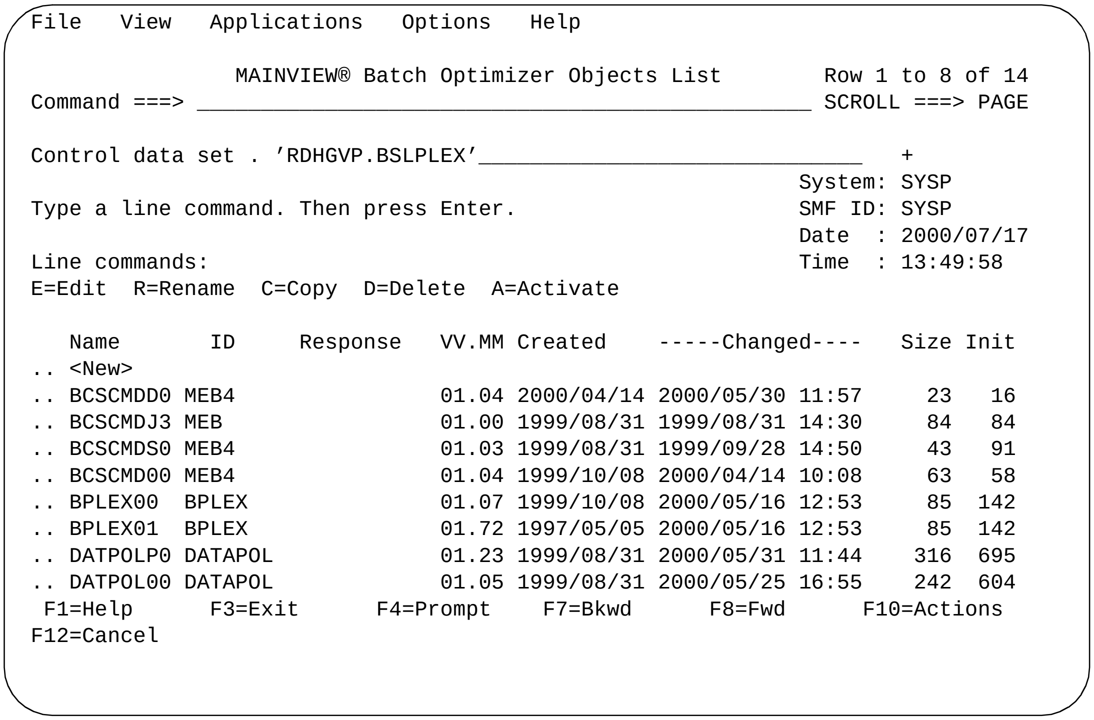
Step 1 Access the MAINVIEW Batch Optimizer Objects List panel (Figure 5-3 on page 5-7). For information on editing or viewing a BatchPlex definition, see the *MAINVIEW Batch Optimizer Installation Manual*.

Step 2 Position the cursor in the action entry field to the left of your choice of UCF definition. Type **E** (Edit), and press **Enter**.

The dialog displays an ISPF member in edit mode. Edit or view the definition, as required.

Step 3 To exit without saving your changes, press **F12** (Cancel). To exit and save your changes, press **F3** (Exit). The dialog displays the MAINVIEW Batch Optimizer Objects List panel (Figure 5-3) with the definition at the top of the scrollable area.

Figure 5-3 MAINVIEW Batch Optimizer Objects List Panel



```
File   View   Applications   Options   Help

                MAINVIEW® Batch Optimizer Objects List        Row 1 to 8 of 14
Command ===> ________________________________________________ SCROLL ===> PAGE

Control data set . 'RDHGVP.BSLPLEX'_____________________________   +
                                                            System: SYSP
Type a line command. Then press Enter.                      SMF ID: SYSP
                                                            Date  : 2000/07/17
Line commands:                                              Time  : 13:49:58
E=Edit  R=Rename  C=Copy  D=Delete  A=Activate

   Name       ID     Response   VV.MM Created    -----Changed----   Size Init
.. <New>
.. BCSCMDD0 MEB4                01.04 2000/04/14 2000/05/30 11:57     23   16
.. BCSCMDJ3 MEB                 01.00 1999/08/31 1999/08/31 14:30     84   84
.. BCSCMDS0 MEB4                01.03 1999/08/31 1999/09/28 14:50     43   91
.. BCSCMD00 MEB4                01.04 1999/10/08 2000/04/14 10:08     63   58
.. BPLEX00  BPLEX               01.07 1999/10/08 2000/05/16 12:53     85  142
.. BPLEX01  BPLEX               01.72 1997/05/05 2000/05/16 12:53     85  142
.. DATPOLP0 DATAPOL             01.23 1999/08/31 2000/05/31 11:44    316  695
.. DATPOL00 DATAPOL             01.05 1999/08/31 2000/05/25 16:55    242  604
 F1=Help      F3=Exit      F4=Prompt    F7=Bkwd      F8=Fwd      F10=Actions
F12=Cancel
```

Activating UCF Definitions

After you use the MAINVIEW Batch Optimizer dialog to edit a UCF definition, you must activate the definition to have your changes take effect. Simply saving an edited definition does not cause the changes to take effect.

Activating a UCF Definition

Your site’s initial UCF definitions were created during the MAINVIEW Batch Optimizer customization process. They become active when you start the batch performance subsystem on the images in the BatchPlex. The names of the UCF members are defined within the BatchPlex definition.

For a new UCF definition, you must activate the definition on each MVS image in the BatchPlex.

You can use a MAINVIEW Batch Optimizer command or the MAINVIEW Batch Optimizer dialog to activate a UCF definition. Using the commands or dialog, however, activates the definition for the current activation of the BatchPlex only. When you restart the batch performance subsystem the UCF definition that is identified in the BatchPlex definition become active again.

To make a UCF definition permanent, you must change the UCF definition name in the BatchPlex definition.

When you activate a UCF definition using a command or the dialog, the batch performance subsystem responds with messages such as the following to indicate whether the activation was successful:

- UCF definition was activated successfully.
- UCF definition was not found in the control data set.
- UCF definition contains an error. The subsystem does not activate the UCF definition.

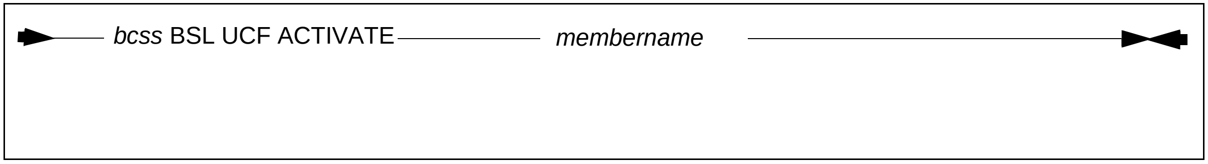
BMC Software, Inc., Confidential and Proprietary Information

Activating a UCF Definition Using a Command

To activate a UCF definition using a command, enter the BSL UCF ACTIVATE command from the MVS console.

The syntax of this command is as follows:

Figure 5-4 BSL UCF Activate



where bcss is the ID of the batch performance subsystem that is active on the MVS image, and membername is the name of an existing UCF definition in the control data set.

Activating a UCF Definition Using the Dialog

To activate a UCF definition using the dialog, complete the following steps:

Step 1 Access the MAINVIEW Batch Optimizer Objects List panel Figure 5-3 on page 5-7. For information on editing or viewing a BatchPlex definition, see the *MAINVIEW Batch Optimizer Installation Manual*.

Step 2 Position the cursor in the action entry field to the left of UCF definition. Type **A** (Activate), and press **Enter**.

The dialog displays the Policy Activation BatchPlex Name pop-up.

Step 3 Type the name of a BatchPlex on which to activate the policy or definition, and press **Enter**.

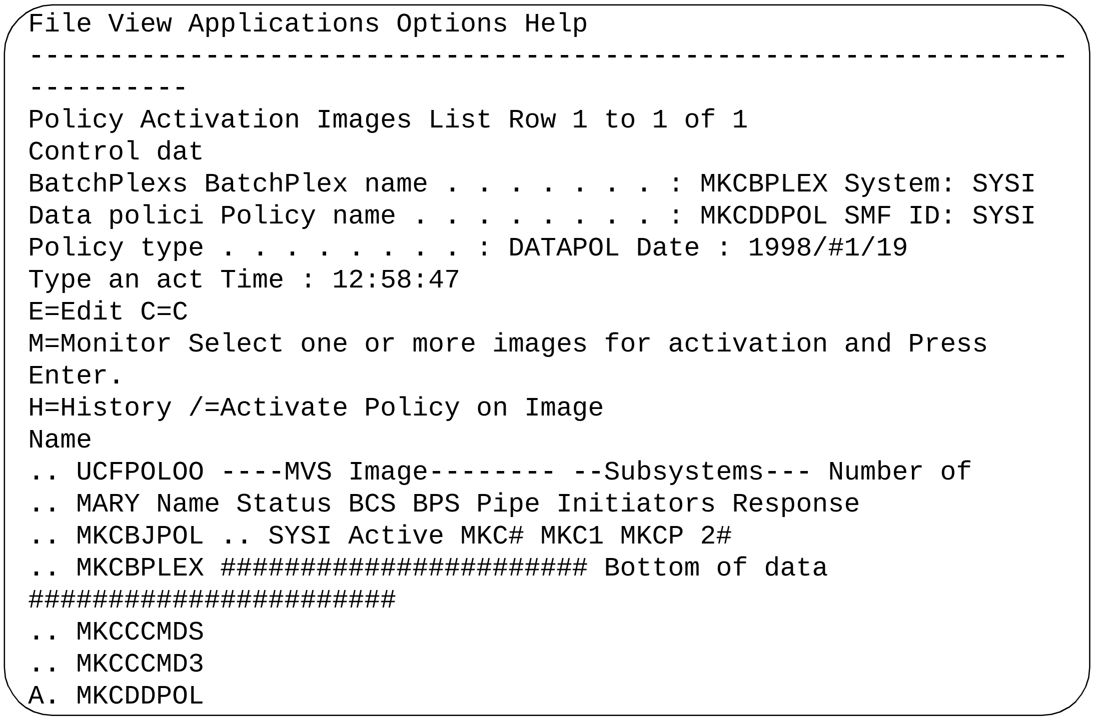
You can view a selection list of item types. Position the cursor in the BatchPlex name field, and press **F4** (Prompt).

The dialog displays a BatchPlex Definition List pop-up, which lists the available BatchPlexes. To select an item from the list, position the cursor to the left of a name, and press **Enter**. The dialog displays the Policy Activation BatchPlex Name pop-up with your choice of BatchPlex displayed in the BatchPlex name field. If you do not want to select an item type from the list, press **F12** (Cancel). The dialog redisplays the Policy Activation BatchPlex Name pop-up.

Note: The BatchPlex you specify must be active on the same MVS image as your dialog session and must be defined in the control data set allocated to your dialog session.

The dialog displays the Policy Activation Images List pop-up.

Figure 5-5 Policy Activation Images List Pop-Up



```
File View Applications Options Help
-----------------------------------------------------------------
----------
Policy Activation Images List Row 1 to 1 of 1
Control dat
BatchPlexs BatchPlex name . . . . . . . . : MKCBPLEX System: SYSI
Data polici Policy name . . . . . . . . . : MKCDDPOL SMF ID: SYSI
Policy type . . . . . . . . . : DATAPOL Date : 1998/#1/19
Type an act Time : 12:58:47
E=Edit C=C
M=Monitor Select one or more images for activation and Press
Enter.
H=History /=Activate Policy on Image
Name
.. UCFPOLOO ----MVS Image-------- --Subsystems--- Number of
.. MARY Name Status BCS BPS Pipe Initiators Response
.. MKCBJPOL .. SYSI Active MKC# MKC1 MKCP 2#
.. MKCBPLEX ####################### Bottom of data
#######################
.. MKCCCMDS
.. MKCCCMD3
A. MKCDDPOL
```

Step 4 Position the cursor in the action entry field to the left of each MVS Image in the BatchPlex on which to activate the policy or definition. Type **/** (Activate Policy on Image) next to each applicable MVS Image, and press **Enter**.

The Response field for each of the selected MVS images indicates if the policy or definition was successfully activated.

Step 5 Press **F12** (Cancel) or **F3** (Exit) to return to the MAINVIEW Batch Optimizer Objects List panel.

BMC Software, Inc., Confidential and Proprietary Information

Determining Which Policy or UCF Definition Is Active

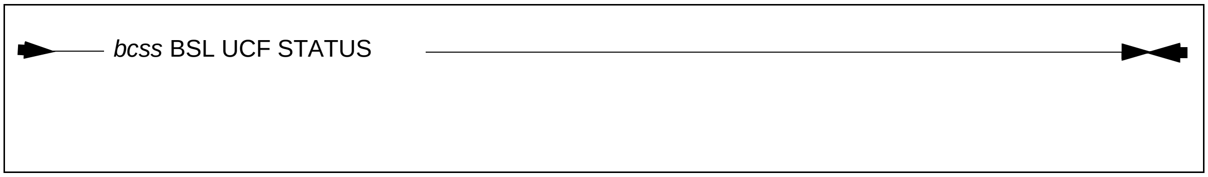
Because a new job policy or UCF definition can be activated at any time, the policy or UCF definition that is identified in the BatchPlex definition might not be the one that is in effect. Therefore, before making changes to a policy or UCF definition based on current activities in your installation, you should determine which definitions are active.

Determining Which UCF Definition is Active

To determine which UCF definition is active, enter the BSL UCF STATUS command from the MVS console.

The syntax of this command is:

Figure 5-6 BSL UCF Status



where bcss is the ID of the MAINVIEW Batch Optimizer subsystem that is active on the MVS image.

The MAINVIEW Batch Optimizer subsystem responds with messages that indicate the name of the active UCF definition. Also, the active BatchPlex definition is displayed.

UCF Options

This section describes the purpose of each option in a UCF definition. UCF definition options let you define Job Optimizer behavior when processing various objects in your environment. The following sections discuss the various types of objects

- DDName objects
- device objects
- event objects
- program objects

DDname Object Options

When specifying a Data Definition (DD) name object, JCL naming conventions apply. Job Optimizer will process the DD name specified for this option in the manner specified by the UCF statement. Job Optimizer will take the action specified on the statement when a job being processed by Job Optimizer references this DD name.

To define a DD name object, you must specify the following information:

- DD name
- an action to perform when Job Optimizer detects this DD name
- a target BatchPlex eligible image if you specify the TARGET action

DDName Object Option

In this field, specify the name of a DD name that requires special handling on the part of Job Optimizer. The name must follow JCL naming conventions for DD names.

Default: None

Policy Keyword: DDNAME={1–8 character DD name}

Action Option

With this option, you can indicate how Job Optimizer is to process the specified object (in this case, a DD name) when encountered while attempting to optimize a job. For DD name objects, the valid actions are as listed in Table 5-1.

Table 5-1 Valid Actions for DD Name Objects (Part 1 of 2)

Action	Description
TARGET	Execute steps containing this DD name on a specific MVS image in the active BatchPlex.
SPLIT	Specifies that steps containing this DD name are to be split regardless of any Job Optimizer determined constraints. Use the SPLIT action with extreme caution.
BYPASS	Specifies that steps containing this DD name are not to be split.
JOBBYPASS	Specifies that jobs containing this DD name or executing this program are not to be split.

Table 5-1 Valid Actions for DD Name Objects (Part 2 of 2)

Action	Description
RESTART	Specifies that steps containing this DD name are part of a restart package. Only the first step of a job stream is checked against the UCF policy for this action. Job Optimizer will not do any analysis or schedule any splitting until after the first step is complete.
SERIALIZE	Specifies that steps containing this DD name are to be serialized in relationship to other steps. That is, all steps prior to this step must complete prior to this step beginning and no subsequent steps may begin execution until after this step is completed. Note: This does not affect targeting of the step.
NOSERIALIZE	Specifies that steps containing this DD name should not be serialized. This is the default.
SERIALIZECC	combination of SERIALIZE and JESCC actions Specifies that this step is to be serialized in relationship to other steps. That is, all steps prior to this step must complete prior to this step beginning, and no subsequent steps may begin execution until after this step is completed. The step will run in the JES initiator, and condition codes on the EXEC statement will be honored. This option will override the job BSLCNTL NOCC option
EXPEDITE	Specifies that steps containing this DD name will not use the piping subsystem. This option only applies to steps executing in a Job Optimizer extended job entry subsystem (XJS) initiator. Steps executing within a Job Optimizer XJS initiator, when complete, will free the XJS initiator. If the job policy global option or definition performance option has the *Sequential step termination option*' set to **Y**, the step processes as if the *Sequential step termination option* is set to **N**. As steps terminate and free the XJS initiators, Job Optimizer satisfies the data set dispositions specified within the JCL and filed updates are also performed at the completion of the job step.
JESCC	Specifies that steps containing this DD name are not to be split. The step will run in the JES initiator and condition codes on the EXEC statement will be honored. This option will override the job BSLCNTL NOCC option.
NOOP	Specifies that this is a place holder entry. No action will take place.

Default: SERIALIZE

Policy Keyword:

ACTION={BYPASS|EXPEDITE|JESCC|JOBBYPASS|NOOP|NOSERIALIZE|RESTART|SERIALIZE|SERIALIZECC|SPLIT|TARGET}

MVS System Name Option

Used with an action of **TARGET**, this option identifies the target MVS image for steps that contain this DD name. The MVS system name must be one that is part of the active BatchPlex to be eligible for step Targeting. MVS system naming conventions apply. Job Optimizer honors this option only when Target is the specified action.

Default: None

Policy Keyword: SYSTEM={any eligible BatchPlex MVS image}

Device Object Option

Use the Device Object option to describe site-specific allocation esoteric information to Job Optimizer. Job Optimizer recognizes certain standard device type specifications such as TAPE, SYSDA, and VIO. If you use more site-specific esoteric names, you must provide device information via the Device object statements. To inform Job Optimizer of these unit name allocation esoteric names, specify:

To define a Device object, you must specify the following items:

- device name
- type of device represented by the name
- action to perform when Job Optimizer detects this device
- target BatchPlex eligible image if you specify the TARGET action

Device Object

With this option, specify the device esoteric name that you want to define to Job Optimizer.

Default: None

Policy Keyword: DEVICE={any valid esoteric unit name}

Device Type Option

With this option, you can indicate the actual device type reflected by the esoteric. This option has the following possible values:

Table 5-2 Valid Values for the Device Type Option

Action	Description
DISK	Indicates that Job Optimizer should treat allocations to this object as direct access storage device (DASD) requests.
TAPE	Indicates that Job Optimizer should treat allocations to this object as tape requests.
VIO	Indicates that Job Optimizer should treat allocations to this object as VIO requests.

Default: None

Policy Keyword: DEVTYPE={DISK|TAPE|VIO}

Action

With this option, you can optionally indicate how Job Optimizer is to process the specified device type when encountered while attempting to optimize a job. For Device objects, the following are valid actions:

Table 5-3 Valid Values for the Action Option

Action	Description
TARGET	Execute steps access this device type on a specific MVS image in the active BatchPlex.
BYPASS	Specifies that steps accessing this device type are not to be split.

Default: SERIALIZE

Policy Keyword: ACTION={BYPASS|TARGET}

MVS System Name Option

Used in conjunction with an action of **TARGET**, this option identifies the target MVS image for steps that accessing this device type. The MVS system name must be one that is part of the active BatchPlex to be eligible for step Targeting. MVS system naming conventions apply. Job Optimizer honors this option only when Target is the specified action.

Default: None

Policy Keyword: SYSTEM={any eligible BatchPlex MVS image}

Event Object Option

The Event Object option specification allows you to alter how Job Optimizer will process certain allocation events. Job Optimizer will process these events in the manner specified by this UCF statement. The type of event can either be an esoteric unit name (UNIT) or an SMS storage group (STORGRP) specification. When Job Optimizer processes a job that performs allocations defined by an Event object statement, it will change the allocation information to match the data in the object.

To define an Event object, you must specify the following items:

- type of allocation event to process
- name of the allocation event to be altered
- new name for the allocation event

Event Type Option

With the Event Type option, you can indicate the type of allocation event you wish to affect. The Table 5-4 describes the valid options:

Table 5-4 Valid Values for the Event Type Option

Value	Description
STORGRP	The event is an SMS storage group allocation. When Job Optimizer detects an allocation event of this type and the name of the storage group matches the value specified for the **TO** option, it changes the storage group name to the one specified for the **FROM** option. storage group name to the one specified for the **FROM** option.
UNIT	The event is an esoteric unit name allocation. When Job Optimizer detects an allocation event of this type and the esoteric unit name matches the value specified for the **TO** option, it changes the esoteric unit name to the one specified for the **FROM** option.

Default: None

Policy Keyword: EVENTID={STORGRP|UNIT}

From Option

With this option, specify the object that Job Optimizer will replace when encountered during allocation. Job Optimizer will replace this value with the value specified for the **To** option. If you specify STORGRP as the Event type, Job Optimizer treats this value as an SMS storage group. If you specify UNIT as the Event type, Job Optimizer treats this value as an esoteric unit name.

Default: None

Policy Keyword: FROM={any valid SMS Storage group or esoteric unit name}

To Option

With this option, specify the value that Job Optimizer will use as a replacement for the value specified for the **From** option during an allocation event. If you specify STORGRP as the Event type, Job Optimizer treats this value as an SMS storage group. If you specify UNIT as the Event type, Job Optimizer treats this value as an esoteric unit name.

Default: None

Policy Keyword: TO={any valid SMS Storage group or esoteric unit name}

BMC Software, Inc., Confidential and Proprietary Information

Program Object

When specifying a Program object, JCL naming conventions apply. Job Optimizer will process the program specified for this option in the manner specified by the UCF statement. Job Optimizer will take the action specified on the statement when a job being processed by Job Optimizer executes this program.

To define a Program object, you must specify the following information:

- program name
- action to perform when Job Optimizer executes this program
- target BatchPlex eligible image if you specify the TARGET action

Program Object Option

In this field, specify the name of a program that requires special handling on the part of Job Optimizer. The name must follow data set member naming conventions for programs.

Default: None

Policy Keyword: PROGRAM={1–8 character program name}

Action

With this option, you can indicate how Job Optimizer is to process the specified object (in this case, a program name) when encountered while attempting to optimize a job. For program objects, the valid actions are described in Table 5-5:

Table 5-5 Valid Values for Program Object Action Option (Part 1 of 2)

Action	Description
TARGET	Execute steps executing this program on a specific MVS image in the active BatchPlex.
SPLIT	Specifies that steps executing this program are to be split regardless of any Job Optimizer determined constraints. Use the SPLIT action with extreme caution.
BYPASS	Specifies that steps executing this program are not to be split.
JOBBYPASS	Specifies that jobs executing this program or executing this program are not to be split.
RESTART	Specifies that jobs executing this program or executing this program are not to be split.

Table 5-5 Valid Values for Program Object Action Option (Part 2 of 2)

Action	Description
SERIALIZE	Specifies that steps executing this program are to be serialized in relationship to other steps. That is, all steps prior to this step must complete prior to this step beginning and no subsequent steps may begin execution until after this step is completed. **Note:** This does not affect targeting of the step.
NOSERIALIZE	Specifies that steps executing this program should not be serialized. This is the default.
SERIALIZECC	combination of SERIALIZE and JESCC actions Specifies that this step is to be serialized in relationship to other steps. That is, all steps prior to this step must complete prior to this step beginning, and no subsequent steps may begin execution until after this step is completed. The step will run in the JES initiator, and condition codes on the EXEC statement will be honored. This option will override the job BSLCNTL NOCC option.

Default: SERIALIZE

Policy Keyword: ACTION={BYPASS|JOBBYPASS|NOSERIALIZE|RESTART|SERIALIZE|SERIALIZECC|SPLIT|TARGET}

MVS System Name

Used in conjunction with an action of **TARGET**, this option identifies the target MVS image for steps that execute this program. The MVS system name must be one that is part of the active BatchPlex to be eligible for step Targeting. MVS system naming conventions apply. Job Optimizer honors this option only when Target is the specified action.

Default: None

Policy Keyword: SYSTEM={any eligible BatchPlex MVS image}

BMC Software, Inc., Confidential and Proprietary Information

Chapter 6 Job Optimizer Commands

This chapter describes the commands that are available in Job Optimizer. This chapter discusses the following topics:

Summary of Job Optimizer Commands

Table 6-1 lists the commands that you use to control the MAINVIEW® Batch Optimizer components and subsystems.

Table 6-1 MAINVIEW Batch Optimizer Components and Subsystems Commands (Part 1 of 2)

Command	Function	Required RACF Access
BMC Software Primary Subsystem		
START bmcpproc	starts the BMC Software Primary Subsystem	CONTROL
bmcp STATUS	provides the subsystem address space ID and indicates whether DEBUG mode is active or inactive	READ
bmcp SHUTDOWN	offers a nondisruptive way of stopping a BMC Software Primary subsystem, allowing jobs to finish before ending the subsystem	CONTROL
CANCEL bmcpproc	stops the BMC Software Primary Subsystem immediately, regardless of whether work is in progress	CONTROL
MAINVIEW Batch Optimizer Subsystem		
START mbosproc	starts the MAINVIEW Batch Optimizer Subsystem	CONTROL
mbos STATUS	provides the subsystem address space ID and indicates whether DEBUG mode is active or inactive	READ
mbos SHUTDOWN	stops the MAINVIEW Batch Optimizer Subsystem and stops all processing for Job Optimizer and Data Optimizer	CONTROL
CANCEL mbosproc	stops the MAINVIEW Batch Optimizer Subsystem immediately, regardless of whether work is in progress	CONTROL
Extended Job Execution Subsystem		
mbos BSL XJS STARTUP	starts the extended job execution subsystem and the MBOX initiators	CONTROL
mbos BSL XJS STATUS	provides information about the extended job execution subsystem and indicates whether the MBOX initiators are active or inactive	CONTROL
mbos BSL XJS QUIESCE	stops the MBOX initiators on a particular MVS image	CONTROL
mbos BSL XJS ACTIVATE	starts the MBOX initiators on a particular MVS image	CONTROL
mbos BSL XJS SHUTDOWN	stops the extended job execution subsystem and the MBOX initiators	CONTROL
mbos BSL XJS TRACE	provides trace information that BMC Software can use to follow the processing of the extended job execution subsystem	CONTROL

BMC Software, Inc., Confidential and Proprietary Information

Command	Function	Required RACF Access
Job Optimizer Component		
mbos REINIT BSL	reinitializes Job Optimizer after an IPL or abend	CONTROL
mbos BSL STATUS	provides general information about the status of Job Optimizer	CONTROL
mbos BSL DISABLE	stops the Job Optimizer component from intercepting jobs for performance processing	CONTROL
mbos BSL ENABLE	starts the Job Optimizer component, which can begin intercepting jobs for performance processing	CONTROL
mbos BSL QUIESCE	stops a BatchPlex image from accepting Job Optimizer work	CONTROL
mbos BSL WAITS	displays the waiting status of job steps that are running in an MBOX initiator	CONTROL
mbos BSL POLICY ACTIVATE	activates a new or changed job policy	CONTROL
mbos BSL POLICY STATUS	displays the active job policy definition	CONTROL
mbos POLICY DISPLAY	is where Job Optimizer is to write the job policy action definition that is associated with a job upon termination of the job	CONTROL
mbos BSL READJTL	sets or displays the default action for the READJTL job policy action definition	CONTROL
mbos BSL UCF STATUS	displays the active UCF definition	CONTROL
mbos BSL UCF Activate	activates a new or changed UCF definition	CONTROL
mbos BSL GROUP	specifies which JES3 initiator groups MAINVIEW Batch Optimizer is to intercept and lists the initiator groups that MAINVIEW Batch Optimizer is intercepting	CONTROL
mbos BSL DUMP	produces various dumps that pertain to the specified batch performance subsystem	CONTROL
mbos BSL DEBUG	controls the capturing of dumps for the specified batch performance subsystem	CONTROL
Job Optimizer Pipes Component		
START mbop	starts Job Optimizer Pipes	CONTROL
F mbop, SHUT TYPE=NORMAL	stops Job Optimizer Pipes	CONTROL
Data Optimizer Component		
mbos REINIT DAP	reinitializes Data Optimizer after an IPL or abend	CONTROL
mbos DAP STATUS	provides general information about the status of Data Optimizer	CONTROL
mbos DAP DISABLE	stops the Data Optimizer component from providing performance processing for data set I/O requests	CONTROL
mbos DAP ENABLE	starts the Data Optimizer component, which begins providing performance processing for data set I/O requests	CONTROL

BMC Software, Inc., Confidential and Proprietary Information

Command	Function	Required RACF Access
mbos DAP HISTKEY DSNPGM	creates the history key, where the history key is DNS and program name	CONTROL
mbos DAP HISTKEY FULL	creates the history key, where the history key is job, step, procstep, program and DDNAME	CONTROL
mbos DAP POLICY STATUS	displays the active data policy definition	CONTROL
mbos DAP POLICY ACTIVATE	activates a new or changed data policy	CONTROL

BMC Software, Inc., Confidential and Proprietary Information

Starting the BMC Software Primary Subsystem

Summary: The BMC Software Primary Subsystem (BMCP) is required by the MAINVIEW Batch Optimizer subsystem. If you start the MAINVIEW Batch Optimizer subsystem before starting the BMC Software Primary Subsystem, the MAINVIEW Batch Optimizer subsystem detects that the BMC Software Primary Subsystem is not available. The BMC Software Primary Subsystem then enters its own command to start the BMC Software Primary Subsystem.

To start the BMC Software Primary Subsystem, complete the following steps:

Step 1 *Required*. Issue the following command:

```
START  bmcpproc
```

Step 2 *Optional*. Check the system log to ensure the command successfully completed. The BMC Software Primary Subsystem responds with messages indicating its availability. Status messages are also displayed.

Displaying the BMC Software Primary Subsystem Status

Summary: If you begin to see BMCP-prefixed messages in the system log, you can display the status of the BMC Software Primary Subsystem to determine whether the subsystem is active. Each MVS image will have only one BMC Software Primary Subsystem active.

To display the status of the BMC Software Primary Subsystem, complete the following steps:

Step 1 *Required*. Issue the following command:

```
bmcp  STATUS
```

Step 2 *Optional*. Check the system log to ensure the command successfully completed. The BMC Software Primary Subsystem responds with messages indicating the status of the BMC Software Primary Subsystem.

Stopping the BMC Software Primary Subsystem

Summary: BMC Software recommends that you leave the BMC Software Primary Subsystem running continuously. You must stop it, however, as part of an orderly shutdown of MVS before an IPL. To stop the BMC Software Primary Subsystem in a nondisruptive way, issue the SHUTDOWN command.

To stop the BMC Software Primary Subsystem, complete the following steps:

Step 1 *Required*. Issue the following command:

```
bmcpproc  SHUTDOWN
```

Step 2 *Optional*. Check the system log to ensure the command successfully completed. The BMC Software Primary Subsystem responds with messages indicating that termination was initiated and completed. Status messages are also displayed.

Canceling the BMC Software Primary Subsystem

Summary: To stop the BMC Software Primary Subsystem immediately, use the MVS CANCEL command.

To cancel the BMC Software Primary Subsystem, issue the following command:

```
SHUTDOWN  bmcpasid
```

asid is the address space ID of the BMC Software Primary Subsystem.

Starting the MAINVIEW Batch Optimizer Subsystem

Summary: You must start the MAINVIEW Batch Optimizer subsystem before any Job Optimizer processing can occur. Start the MAINVIEW Batch Optimizer subsystem with new copies of resident MAINVIEW Batch Optimizer subsystem modules by entering the MVS START command.

To start the MAINVIEW Batch Optimizer subsystem, issue the following command:

```
START  mbobcss  ,AUTOSTART=x
```

Include the AUTOSTRT=Y parameter to have the MAINVIEW Batch Optimizer subsystem to use the autostart specifications for MVS images in the BatchPlex definition. MVS START commands are routed to all MVS images associated with the BatchPlex for which the **Automatically Start Subsystems** field is set to Y for the image.

Specify the AUTOSTRT=N parameter to have the MAINVIEW Batch Optimizer subsystem ignore the autostart specifications for MVS images in the BatchPlex definition.

The subsystems are not automatically started on other MVS images in the BatchPlex.

Effects of Starting the MAINVIEW Batch Optimizer Subsystem

When you start the MAINVIEW Batch Optimizer subsystem, the following occur:

- The MAINVIEW Batch Optimizer subsystem responds with messages indicating its availability.
- If the BMC Software Primary Subsystem is not active, the MAINVIEW Batch Optimizer subsystem starts them automatically.

If you have not performed an IPL since the last time you started the MAINVIEW Batch Optimizer subsystem, the BMC Software Primary Subsystem is not restarted. The BMC Software Primary Subsystem is still in memory from the previous start.

The BatchPlex that is identified in the MAINVIEW Batch Optimizer subsystem procedure is started, and the job policy and UCF member that are identified in the BatchPlex definition are activated.

If you previously used the ACTIVATE command to start another job policy or UCF member, these policies are not refreshed from memory when the MAINVIEW Batch Optimizer subsystem is started. Only the job policy and UCF member that are identified in the BatchPlex definition are started. To ensure that the correct job policy and UCF member are activated when the MAINVIEW Batch Optimizer subsystem is started, you must change the BatchPlex definition.

Placing the START Command

You can place the MVS START command for the MAINVIEW Batch Optimizer subsystem in a COMMND*xx* member of PARMLIB so that the subsystem starts automatically during an IPL. Do not include the AUTOSTRT parameter on START commands that are placed in a COMMNDxx member of PARMLIB.

Starting Under an Alternate JES Subsystem

If your installation uses alternate JES subsystems, you can initialize Job Optimizer under an alternate JES subsystem by using the SUB parameter in the START command of the MAINVIEW Batch Optimizer subsystem. For example:

```
s mbosproc,SUB=altj
```

where:

- *mbosproc* is the procedure name for the MAINVIEW Batch Optimizer subsystem.
- *altj* is the subsystem ID of the alternate JES subsystem.

When Job Optimizer is initialized under the alternate JES subsystem, Job Optimizer intercepts only jobs that run under the alternate JES subsystem.

Displaying the MAINVIEW Batch Optimizer Subsystem Status

Summary: You can display the status of the MAINVIEW Batch Optimizer subsystem to determine whether the subsystem is active. Display the status of the MAINVIEW Batch Optimizer subsystem by entering the STATUS command.

To display the status of the MAINVIEW Batch Optimizer subsystem, complete the following steps:

Step 1 *Required*. Issue the following command:

```
mbos  STATUS
```

Step 2 *Optional*. Check the system log to ensure the command successfully completed. The MAINVIEW Batch Optimizer subsystem responds with messages indicating the status of the address space.

Stopping the MAINVIEW Batch Optimizer Subsystem

Summary: You can leave the MAINVIEW Batch Optimizer subsystem running continuously. You must stop it, however, as part of an orderly shutdown of MVS before an IPL. To stop the MAINVIEW Batch Optimizer subsystem address space in a nondisruptive manner, issue the SHUTDOWN command.

To stop the MAINVIEW Batch Optimizer subsystem, issue the following command:

```
mbos  SHUTDOWN
```

When you stop the MAINVIEW Batch Optimizer subsystem, the following occur:

- The MAINVIEW Batch Optimizer subsystem responds with messages indicating that termination was initiated and completed.

- If Job Optimizer has work executing on the subsystem at the time the SHUTDOWN command is entered, the MAINVIEW Batch Optimizer subsystem issues the BSL QUIESCE NOW command to perform an orderly shutdown.See “Quiescing Job Optimizer” on page 6-22 for details about QUIESCE processing.

- The MAINVIEW Batch Optimizer subsystem remains in memory until you perform an IPL.

Include the optional FORCE command to stop the MAINVIEW Batch Optimizer subsystem address space immediately regardless of any processing that might be occurring when the command is entered.

```
mbos  SHUTDOWN FORCE
```

The MAINVIEW Batch Optimizer subsystem responds with messages indicating that a forced shutdown may cause unpredictable results. You are then given three options:

- U - Continue with the forced shutdown.

- N - Switch to normal shutdown processing. The MAINVIEW Batch Optimizer subsystem issues the BSL QUIESCE NOW command to perform an orderly shutdown. For details about QUIESCE processing, see “Quiescing Job Optimizer” on page 6-22

- Any other key - Stop the forced shutdown. Continue processing as if the command had not been issued.

Canceling the MAINVIEW Batch Optimizer Subsystem

Summary: To stop the MAINVIEW Batch Optimizer subsystem immediately, use the MVS CANCEL command.

To cancel the MAINVIEW Batch Optimizer subsystem, issue the following command:

```
mbos  SHUTDOWN
```

asid is the address space ID of the MAINVIEW Batch Optimizer subsystem.

To determine the address space ID of the MAINVIEW Batch Optimizer subsystem, you can enter the MVS DISPLAY command. The syntax is as follows:

```
DISPLAY ACTIVE,LIST
```

An active MAINVIEW Batch Optimizer subsystem appears in this list, but the step name for this MAINVIEW Batch Optimizer subsystem is modified to reflect its address space ID.

Note: BMC Software recommends that you use the SHUTDOWN command to stop the MAINVIEW Batch Optimizer subsystem in a nondisruptive way. If you use the CANCEL command, unpredictable results might occur.

Warning! Use the MVS FORCE command only as a last resort when the CANCEL command fails to perform its function even after you have entered it several times. Using the FORCE command might require you to re-IPL the system.

Starting the Extended Job Execution Subsystem

Summary: The extended job execution subsystem starts automatically when the MAINVIEW Batch Optimizer subsystem starts. If you must end the extended job execution subsystem, or the subsystem abends, you must restart it manually.

Start the extended job execution subsystem by entering the BSL XJS STARTUP command.

To start the extended job execution subsystem, complete the following steps:

Step 1 *Required*. Issue the following command:

```
mbos  BSL XJS STARTUP
```

Step 2 *Optional*. Check the system log to ensure that the command completed successfully. The MAINVIEW Batch Optimizer subsystem responds with messages indicating that the extended job execution subsystem startup has been initiated and provides information about the XJS initiators being started.

Displaying the Status of the Extended Job Execution Subsystem

Summary: You can display the status of the extended job execution subsystem. This status information includes the number of active extended job execution subsystems within an XCF group and information about each job step initiator, which includes the ASID of the XJS initiator address space. Display the status of the extended job execution subsystem by entering the BSL XJS STATUS command.

To display the status of the extended job execution subsystem, complete the following steps:

Step 1 *Required*. Issue the following command:

```
mbos  BSL XJS STATUS
```

Step 2 *Optional*. Check the system log to ensure the command completed successfully. The MAINVIEW Batch Optimizer subsystem responds with messages indicating the status of the extended job execution subsystem.

Quiescing the Extended Job Execution Subsystem

Summary: The BSL XJS QUIESCE command stops the extended job execution subsystem initiators on a particular MVS image.

To stop the extended job execution subsystem initiators on a particular MVS image, complete the following steps:

Step 1 *Required*. Issue the following command:

```
mbos  BSL XJS QUIESCE
```

Step 2 *Optional*. Check the system log to ensure that the command completed successfully. The MAINVIEW Batch Optimizer subsystem responds with messages indicating that termination was initiated and completed.

Stopping the Extended Job Execution Subsystem

Summary: The SHUTDOWN command for the MAINVIEW Batch Optimizer subsystem will stop the extended job execution subsystem and all XJS initiators as part of its processing. You should leave the extended job execution subsystem running always; however, you might need to stop it if requested by BMC Software Product Support. Stop the extended job execution subsystem and all XJS initiators by entering the BSL XJS SHUTDOWN command.

To stop the extended job execution subsystem, complete the following steps:

Step 1 *Required*. Issue the following command:

```
mbos  BSL XJS SHUTDOWN
```

Step 2 *Optional*. Check the system log to ensure that the command completed successfully. The MAINVIEW Batch Optimizer subsystem responds with messages indicating that termination was initiated and completed.

Reinitializing Job Optimizer

Summary: You must reinitialize Job Optimizer after an IPL, after a Job Optimizer abend, or after maintenance is applied. When you reinitialize Job Optimizer, you reload the Job Optimizer modules in memory. The REINIT BSL command does not refresh the current job policy in memory.

To reinitialize Job Optimizer, complete the following steps:

Step 1 *Required*. Issue the following command:

```
mbos  REINIT BSL
```

Step 2 *Optional*. Check the system log to ensure the command successfully completed. The subsystem responds with messages indicating that it has reinitialized Job Optimizer. Status messages are also displayed.

Displaying the Status of Job Optimizer

Summary: You can display status information about Job Optimizer. This status information provides general information about Job Optimizer, such as version and release information, and the version, release, and maintenance level of Job Optimizer.

To display status information, complete the following steps:

Step 1 *Required*. Issue the following command:

```
mbos  BSL STATUS
```

Step 2 *Optional*. The subsystem responds with messages indicating the status of Job Optimizer.

BMC Software, Inc., Confidential and Proprietary Information

Enabling and Disabling Job Optimizer

Summary: Job Optimizer is enabled when the MAINVIEW Batch Optimizer subsystem is initialized. When enabled, Job Optimizer intercepts job steps as indicated by the job policy definition of the active BatchPlex. If necessary, you can disable Job Optimizer to stop the intercepts.

To disable Job Optimizer, complete the following steps:

Step 1 *Required*. Issue either of the following commands:

```
mbos  BSL ENABLE
mbos  BSL DISABLE
```

Step 2 *Optional*. Check the system log to ensure the command successfully completed. The subsystem responds with messages indicating that it has disabled Job Optimizer.

Quiescing Job Optimizer

Summary: The BSL QUIESCE command lets you to end Job Optimizer processing on a particular MVS image. Currently running work is allowed to complete without interruption; however, no additional jobs are selected for splitting or targeting on the image.

Enter BSL QUIESCE on each MVS image in which no additional Job Optimizer work should be started. After entering BSL QUIESCE, you should monitor the image until all splitting and targeting on the image is complete.

In addition to quiescing individual images, you can use BSL QUIESCE to complete the orderly shutdown of an entire BatchPlex. Enter BSL QUIESCE on each MVS image in the BatchPlex.

To quiesce Job Optimizer, complete the following steps:

Step 1 *Required*. Issue one of the following commands:

```
mbos  BSL QUIESCE
```

Include the optional NOW parameter to cancel existing jobs rather than allowing the jobs to complete normally. This command allows Job Optimizer to quiesce more quickly.

```
mbos  BSL QUIESCE NOW
```

Step 2 *Optional*. Check the system log to ensure the command successfully completed. The subsystem responds with messages indicating that it has disabled Job Optimizer.

If jobs under the control of Job Optimizer are active, message BMC173156W is displayed to indicate that work is in progress. The message allows you to control how the active work is handled as follows:

- CANCEL - Instructs the system to begin cancelling all jobs that are under the control of Job Optimizer.
- WAIT - Instructs the system to wait until one of two events occurs—a job terminates normally or a five-minute interval expires. In either case, the request to quiesce Job Optimizer is restarted. The processing loop continues until all jobs end or until you change your response to CANCEL or IGNORE.
- IGNORE - Instructs the system to stop QUIESCE NOW processing immediately.

Displaying the Status of Job Optimizer

Summary: The BSL SYSTEMS command displays the status of a system that is participating in a BatchPlex.

To display the status of a system participating in a BatchPlex, complete the following steps:

Step 1 *Required*. Issue the following command:

```
ssid  BSL SYSTEMS
```

Step 2 *Optional*. Check the system log to ensure the command successfully completed. The subsystem responds with a message indicating the status of the system. The message will indicate that the system is either:

- ACTIVE - The system is participating in a BatchPlex
- DEFINED - The system was defined in the BatchPlex but has not initialized.
- INACTIVE - The system connected to the BatchPlex but has since terminated.

Activating a Job Policy

Summary: Your site’s initial job policy was created during the MAINVIEW Batch Optimizer customization process. The job policy becomes active when you start the MAINVIEW Batch Optimizer subsystem on the images in the BatchPlex. The name of the job policy is defined within the BatchPlex definition. For a new job policy, you must activate the policy on each MVS image in the BatchPlex.

To activate a job policy, issue the following command:

```
mbos  BSL POLICY ACTIVATE policyname
```

where:

- *mbos* is the ID of the MAINVIEW Batch Optimizer subsystem that is active on the MVS image

- *policyname* is the name of an existing job policy definition in the control data set

BMC Software, Inc., Confidential and Proprietary Information

Displaying the Status of a Job Policy

Summary: To determine which job policy is active, enter the BSL POLICY STATUS command from the MVS console.

To display the status of a job policy, issue the following command:

```
mbos  BSL POLICY STATUS
```

where *mbos* is the ID of the MAINVIEW Batch Optimizer subsystem that is active on the MVS image

BMC Software, Inc., Confidential and Proprietary Information

Displaying the Waiting Status of Job Steps

Summary: The BSL WAITS command displays the waiting status of job steps running in an XJS initiator.

To display the waiting status of job steps running in an XJS initiator, complete the following steps:

Step 1 *Required*. Issue the following command:

```
mbos  BSL WAITS
```

Step 2 *Optional*. Check the system log to ensure the command successfully completed. The MAINVIEW Batch Optimizer subsystem responds with messages indicating the waiting status.

Determining Where the Definition for a Job is Written

Summary: When a job terminates, the job policy definition that is associated with the job is written. The BSL POLICY DISPLAY command lets you specify where Job Optimizer is to write the job policy definition associated with a job.

To determine where the definition for a job is written, issue the following command:

```
mbos  BSL POLICY DISPLAY option
```

where *mbos* is the ID of the MAINVIEW Batch Optimizer subsystem that is active on the MVS image, and *option* is one of the following:

SYSMSG The job policy definition is written to the job message queue

JOBLOGThe job policy definition is written to the joblog

BOTH The job policy definition is written to both the joblog and the job message queue

NONE The job policy definition is not written

STATUSDisplays the current setting of this command

Setting the Default JTL Action

Summary: The default action taken when the READJTL is specified in the action definition of a job policy is SPLIT. You can change this default by using the BSL READJTL command.

To change the default, issue the following command:

```
mbos  BSL READJTL SPLIT
                  ANALYZE
                  BYPASS
                  STATUS
```

where *mbos* is the ID of the MAINVIEW Batch Optimizer subsystem that is active on the MVS image.

BMC Software, Inc., Confidential and Proprietary Information

Activating a User Control Facility Definition

Summary: To activate a UCF definition by using a command, enter the BSL UCF ACTIVATE command from the MVS console.

To activate a UCF definition, issue the following command:

```
mbos  BSL UCF ACTIVATE membername
```

where *mbos* is the ID of the MAINVIEW Batch Optimizer subsystem that is active on the MVS image and *membername* is the name of an existing UCF definition in the control data set.

Displaying the Active UCF Definition

Summary: To determine which UCF definition is active, enter the BSL UCF STATUS command from the MVS console.

To determine which UCF definition is active, issue the following command:

```
mbos  BSL UCF STATUS
```

where *mbos* is the ID of the MAINVIEW Batch Optimizer subsystem that is active on the MVS image.

The MAINVIEW Batch Optimizer subsystem responds with messages that indicate the name of the active UCF definition. Also, the active BatchPlex definition is displayed.

BMC Software, Inc., Confidential and Proprietary Information

Specifying and Listing Active JES3 Initiator Groups

Summary: JES3 includes the use of named initiator groups. You must specify which initiator groups Job Optimizer to intercept. You specify the initiator group names by using the BSL GROUP command. You can also use the BSL GROUP command to list the initiator groups that Job Optimizer is intercepting.

To specify or list initiator groups, issue the following command:

```
mbos  BSL GROUP ADD  igname
                REMOVE  igname
                LIST
```

where *mbos* is the ID of the MAINVIEW Batch Optimizer subsystem that is active on the MVS image and *igname* specifies the name of the JES3 initiator group

ADDSpecifies to Job Optimizer that it is to intercept the named initiator group

REMOVESpecifies to Job Optimizer that it is not to intercept the named initiator group

The following rules apply to using the BSL GROUP command:

- Use one BSL GROUP command for each initiator group name that you want to specify.

- Use as many BSL GROUP commands as you need to.

- If you are using JES3 initiator groups and do not specify any initiator group names to intercept, Job Optimizer will not intercept any jobs for MAINVIEW Batch Optimizer subsystem processing.

- You can place BSL GROUP commands in the commands member of the control data set. In this way, the JES3 initiator groups Job Optimizer is to intercept are specified automatically during the initialization of the MAINVIEW Batch Optimizer subsystem.

Dumping MAINVIEW Batch Optimizer Subsystem Information

Summary: The BSL DUMP commands control dump information about the MAINVIEW Batch Optimizer subsystem. The information that is dumped varies, depending upon which command you use.

To dump MAINVIEW Batch Optimizer subsystem information, issue the following command:

```
mbos  BSL DUMP
                DEBUG
                JOBnnnn
                JOBID  JOBnnnn
                ASID
```

where *mbos* is the ID of the MAINVIEW Batch Optimizer subsystem that is active on the MVS image and *igname* specifies the name of the JES3 initiator group.

(no parameter)Dumps the address space and the data space of the Batch Optimizer subsystem and the debug data space.

DEBUGDumps the debug data space.

JOBID JOB*nnnn*Dumps the address space and data space of the MAINVIEW Batch Optimizer subsystem, the debug data space, and all JES and XJS address spaces related to the specified job number (*nnnn). The JOBID portion of the parameter is optional.*

ASID*nnnn*Dumps the specified address space.

Capturing MAINVIEW Batch Optimizer Subsystem Information

Summary: The BSL DEBUG commands control the capturing of information about the MAINVIEW Batch Optimizer subsystem. The information that is captured is predetermined by Job Optimizer. The information that is captured is written to a debug data space.

To capture MAINVIEW Batch Optimizer subsystem information, issue the following command:

```
mbos  BSL DEBUG CAPTURE  ON nnnn
                         OFF
                         RESET
```

where *mbos* is the ID of the MAINVIEW Batch Optimizer subsystem that is active on the MVS image.

ON *nnnn*Starts capturing the predetermined information about the Batch Optimizer subsystem. This parameter also specifies the size (in megabytes) of the debug data space.

OFFStops capturing information about the MAINVIEW Batch Optimizer subsystem.

RESETReinitializes the debug data space. The use of this command does not start or stop the capturing of information.

Activating and Terminating MAINVIEW Batch Optimizer Subsystem Tracing

Summary: Job Optimizer provides trace information that BMC Software Product Support can use to follow the processing of the MAINVIEW Batch Optimizer subsystem. Use the BSL TRACE command to record the information; use the BSL DUMP command to produce the trace data that BMC Software Product Support use for analysis.

To record trace information about the MAINVIEW Batch Optimizer subsystem, issue the following command:

```
mbos  BSL TRACE ON
                OFF
```

where *mbos* is the ID of the MAINVIEW Batch Optimizer subsystem that is active on the MVS image.

ONStarts recording MAINVIEW Batch Optimizer subsystem information.

OFFStops recording MAINVIEW Batch Optimizer subsystem information.

Creating a Trace Table

Summary: A trace table is a storage area that Job Optimizer uses to keep trace information about the MAINVIEW Batch Optimizer subsystem. Use the BSL TRTABLE command to create or modify a trace table in the Extended Common Storage Area (ECSA).

To create a trace table, issue the following command:

```
mbos  BSL TRTABLE MODIFY nnnn
                  RESET
                  DELETE
```

where *mbos* is the ID of the MAINVIEW Batch Optimizer subsystem that is active on the MVS image and *nnnn* is the number of entries in the trace table. Valid values are 266 to 65535

MODIFYCreates the table of a given size in ECSA.

RESETClears all entries in the trace table.

DELETEFrees the trace table.

Controlling the Job Optimizer IEFACTRT Exit

Summary: The IEFACTRT command controls the IEFACTRT exit.

To control the IEFACTRT exit, complete the following command:

```
mbos  BSL IEFACTRT ENABLE
                   DISABLE
```

where *mbos* is the ID of the MAINVIEW Batch Optimizer subsystem that is active on the MVS image

ENABLEEnables the IEFACTRT exit.

DISABLEDisables the IEFACTRT exit.

To determine the status of the MAINVIEW Batch Optimizer subsystem IEFACTRT exit, you can enter the MVS DISPLAY command. The syntax is as follows:

```
DISPLAY PROG,EXIT
             ALL
```

Controlling the SCT with Job Optimizer

Summary: The BSL STEPNUM command controls whether the Section Control Table (SCT) for a split step contains step number 1 or should be updated to reflect the actual step number.

To control the SCT, complete the following command:

```
mbos  BSL STEPNUM ON
                  OFF
```

where *mbos* is the ID of the MAINVIEW Batch Optimizer subsystem that is active on the MVS image.

ONReflects the actual step number.

OFFMakes the step number in the SCT a 1.

If you used CONTROL-D to process sysout for an a split step, you must turn STEPNUM ON. BMC Software recommends that you place a BSL STEPNUM ON statement in your BatchPlex command member BCSCMDOO.

BMC Software, Inc., Confidential and Proprietary Information

Chapter 7 Job Transformation Language

This chapter describes the options that are available in Job Optimizer. You must understand the available options to use Job Optimizer effectively. This chapter discusses the following topics:

BMC Software, Inc., Confidential and Proprietary Information

Overview

Job Transformation Language (JTL) lets you control whether certain steps are executed in parallel (split). JTL is activated for a job by using a job policy selection definition whose action definition specifies READJTL for the Action option field.

Scope of JTL Statements

The scope of a JTL statement is determined by its placement in the job stream. JTL statements are valid only when they are placed immediately after a JOB (job level), EXEC PGM= (step level), or DD (DD level) statement. JTL statements that are placed at any other location within the job stream are treated as JCL comment statements (that is, ignored). No JTL statements are supported at the EXEC procedure_name level.

At the job level, the following JTL statements are valid:

- BYPASS
- ANALYZE
- SPLIT (the default action, or as set by the READJTL command)
- NOCC

At the step level, the following JTL statements are valid:

- BYPASS
- SPLIT
- TARGET systemname
- SERIALIZE
- NOSERIALIZE
- SERIALIZECC
- JESCC
- CC
- NOCC

At the DD level, the following JTL statement is valid:

- ACCESS type

You can specify multiple JTL statements at each level. If the statements do not conflict, all of the statements are observed. If the statements conflict, the last JTL statement encountered is used. For example, the SPLIT and NOCC statements conflict. If both statements are specified for a job step, and NOCC is specified second, it will be observed. The SPLIT statement is ignored.

Using Job-Level Statements

A job-level statement must follow immediately after the JOB statement. A job-level statement becomes the default action for all steps in the job. Job-level statements are not required. If you do not supply a job-level statement, the job-level default JTL action is SPLIT, or as set by the READJTL command.

You can use step-level and DD-level statements to override job-level statements. However, a job-level statement of BYPASS or ANALYZE, cannot be overridden. If one of these statements is supplied, the entire job is bypassed or analyzed as specified.

Using Step-Level Statements

A step-level statement must follow immediately after an EXEC PGM=*program_name* statement. A step-level statement affects only the step that precedes it.

Step-level statements are not required. If you do not supply a step-level statement, the step is processed according to the specified job-level statement, the job-level default of SPLIT, or as set by the READJTL command.

Step-level statements do not override the BYPASS or ANALYZE job-level statements. If a step-level JTL statement is detected when the BYPASS or ANALYZE job-level statement is in affect, the step-level statement is ignored (treated as a JCL comment).

Using DD-Level Statements

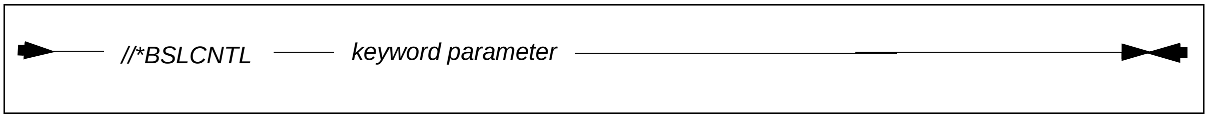
A DD-level statement must follow immediately after a DD statement. A DD-level statement affects only the DD statement that precedes the DD-level statement. DD-level statements are not required. If you do not supply a DD-level statement, Job Optimizer uses information collected in the history data set to determine access intent for the DD statement.

BMC Software, Inc., Confidential and Proprietary Information

JTL Syntax

JTL statements are JCL comment statements. Figure 7-1 shows the syntax.

Figure 7-1 JTL Syntax



Observe the following syntax conventions for JTL statements:

- Each statement must begin with the string //*BSLCNTL. There must be no intervening spaces in this string.
- The //*BSLCNTL string must be followed by one space and a keyword.
- If the keyword has a parameter, there must be one space between the keyword and its parameter.
- Comments can follow the keyword (or parameter) when one space separates the keyword (or parameter) and the comment.

The following examples are valid JTL statements:

```
//BSLCNTL BYPASS

//BSLCNTL ANALYZE

//BSLCNTL SPLIT

//BSLCNTL CC

//BSLCNTL NOCC

//BSLCNTL TARGET SYSP

//BSLCNTL ACCESS OUTPUT

//BSLCNTL SERIALIZE

//BSLCNTL NOSERIALIZE
//BSLCNTL SERIALIZECC

//BSLCNTL JESCC

//BSLCNTL DELHIST

//BSLCNTL SRP

//BSLCNTL JOP
//BSLCNTL NOTAPE
//BSLCNTL TAPE
```

Specifying JTL Statements

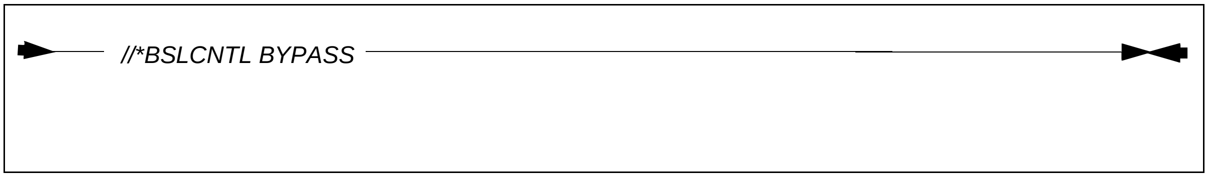
This section describes the following JTL statements:

- BYPASS
- ANALYZE
- SPLIT
- CC
- NOCC
- TARGET
- ACCESS
- SERIALIZE
- NOSERIALIZE
- SERIALIZECC
- DELHIST
- SRP
- JOP
- NOTAPE
- TAPE

Specifying the BYPASS Statement

Figure 7-2 shows the format for the BYPASS statement.

Figure 7-2 BSLCNTL BYPASS



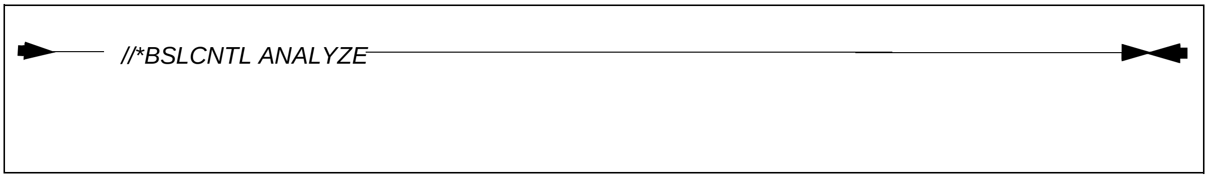
You can specify BYPASS at the job level or at the step level. Step-level and DD-level statements do not override a job-level BYPASS statement. A step-level BYPASS statement overrides a job-level SPLIT statement.

The step or steps within the scope of a BYPASS statement are bypassed and are run in the JES initiator without taking advantage of the ability of the Job Optimizer to dynamically split steps and establish pipes to relieve data constraints.

Specifying the ANALYZE Statement

Figure 7-3 shows the format for the ANALYZE statement.

Figure 7-3 BSLCNTL ANALYZE



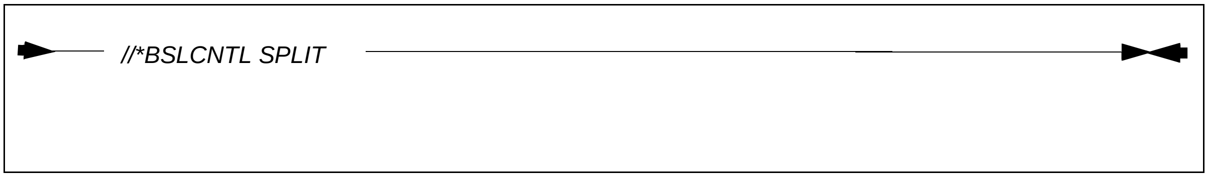
You can specify ANALYZE at the job level only. Step-level and DD-level statements never override a job-level ANALYZE statement.

The steps within the scope of the ANALYZE statement are run in the JES initiator without being split and without pipes being established. Job Optimizer monitors and records data in the history data set that describes how the job runs. With history data, Job Optimizer can effectively and correctly split the job and have pipes established in subsequent runs.

Specifying the SPLIT Statement

Figure 7-4 shows the format for the SPLIT statement.

Figure 7-4 BSLCNTL SPLIT



You can specify SPLIT at the job level or at the step level. Because the default for the READJTL action option in a job policy is SPLIT, specifying a job level SPLIT statement is not required. You can specify a different default by using the READJTL command.

You can specify SPLIT at the step level in combination with the TARGET statement to target steps to selected MVS images.

When you specify a job-level SPLIT statement, Job Optimizer will split the job when all constraints are relieved. Also, when all trust factors have been established, the steps can be directed to run on MVS images that have sufficient capacity to support the workload.

When you specify a step-level SPLIT statement, Job Optimizer will split the job even if constraints have not been relieved.

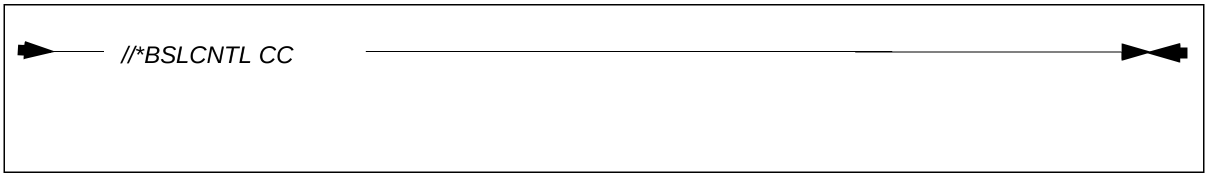
Note: Specifying SPLIT at the step level requires research and testing on your part. If care is not taken, jobs can hang and wait for readers and writers to open pipes that application logic might not require to be open.

Warning! Do not use a step-level SPLIT statement unless you are certain that all constraints have been relieved.

Specifying the CC Statement

Figure 7-5 shows the format for the CC statement.

Figure 7-5 BSLCNTL CC



You can specify a CC statement at the following levels:

- Specify CC at the job level by placing the statement before the first EXEC statement in the job.

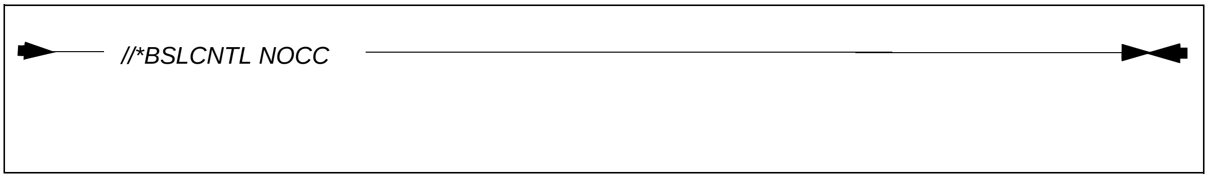
- Specify CC at the step level by placing the statement after the EXEC statement for the job step.

When you include a CC statement, Job Optimizer honors the condition code associated with the job or job step. For a job-level condition code, the step is not split. For a step-level condition, the job step is split after the named job step completes. If the COND= parameter does not name a job step, all previous job steps must complete before the job step is split.

Specifying the NOCC Statement

Figure 7-6 shows the format for the NOCC statement.

Figure 7-6 BSLCNTL NOCC

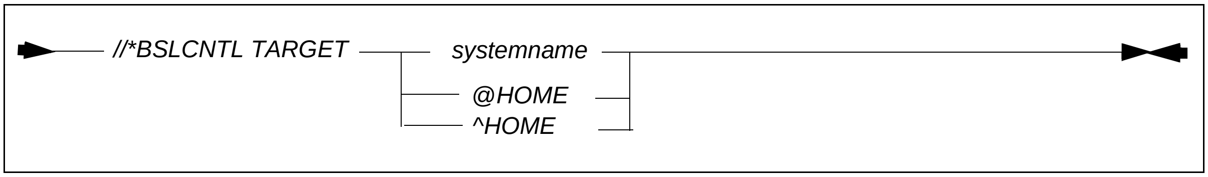


You can specify NOCC at the job level or at the step level. When you include a NOCC statement, Job Optimizer ignores the condition code restraint for a job step. Therefore, Job Optimizer splits the job step if there are no other constraints for the job or job step.

Specifying the TARGET Statement

Figure 7-7 shows the format for the TARGET statement.

Figure 7-7 BSLCNTL TARGET



You can specify TARGET at the step level only. The TARGET statement never overrides a job-level BYPASS or ANALYZE statement. The TARGET statement names the MVS image on which Job Optimizer is to execute the job step if the job step is split. The TARGET statement does not affect whether the job step is split.

You must include one parameter on the TARGET statement, which causes the referenced step to be targeted to run on the MVS image named by the parameter. The parameter is the name of an MVS image (*systemname*) that is included in the BatchPlex definition or the values @HOME, which is the MVS image the job was originally initiated, or ^HOME, which is any valid image in the BatchPlex except the home image.

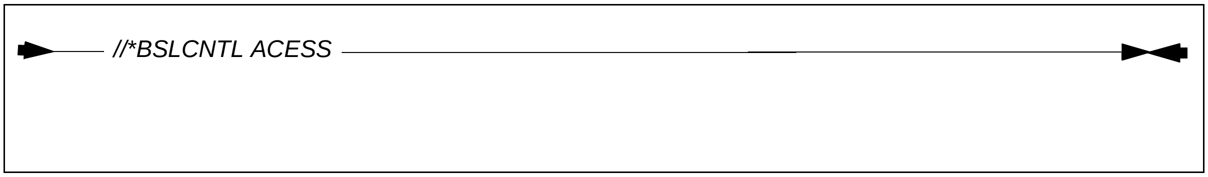
When a TARGET statement overrides a job-level SPLIT statement and a step-level SPLIT statement is not specified, the step is targeted to run on the specified MVS image if all constraints are relieved. If all constraints are not relieved or if the targeted MVS image is not active, the step is not split. This step runs in the JES initiator of the MVS image on which the job was initiated.

When a TARGET statement is used with a step-level SPLIT statement, the step is targeted to the run on the specified MVS image, regardless of whether constraints are relieved. If the MVS image is not active or if Job Optimizer is not running on the named MVS image, the affected step is bypassed and runs in sequence in the JES initiator on the MVS image in which the job was initiated originally.

Specifying the ACCESS Statement

Figure 7-8 shows the format for the ACCESS statement.

Figure 7-8 BSLCNTL ACCESS



You can specify ACCESS at the DD level only. The ACCESS statement instructs Job Optimizer how the immediately preceding DD statement is used in the current run (for input, for output, or for update). Job Optimizer saves this information in the history data set and uses it to establish pipes between writers and subsequent readers of data sets.

If you change a DD statement’s access intent from what is saved in the history data set, Job Optimizer must reevaluate this job before making decisions to execute steps in parallel and establish pipes for the current run.

If you include multiple ACCESS statements in the JCL stream after one DD statement, the last ACCESS statement is processed and saved in the history data set.

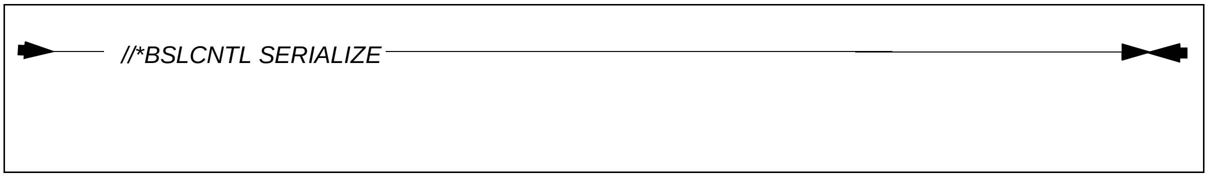
You must include one parameter on the ACCESS statement, which is one of the following values: INPUT, OUTPUT, or UPDATE. The parameter causes the immediately preceding DD statement to be recorded in the history data set as having the type of access in the current run that is listed in the parameter on the ACCESS statement.

Note: You are responsible for properly recording the type of ACCESS that will be made with the referenced DD statement. The ability of the Job Optimizer to properly execute steps in parallel and establish pipes is affected by the accuracy of the data in the history data set.

Specifying the SERIALIZE Statement

Figure 7-9 shows the format for the SERIALIZE statement.

Figure 7-9 BSLCNTL SERIALIZE



You can specify SERIALIZE at the step level. When a step is flagged with the SERIALIZE statement, Job Optimizer runs the step only after all previous steps have completed. Job Optimizer then waits for the serialized step to complete before running all future steps.

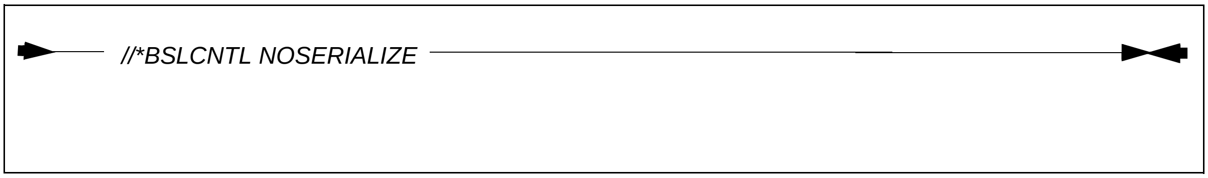
Because JTL statements override User Control Facility (UCF) statements, you can use the SERIALIZE JTL statement to override a NOSERIALIZE statement in the UCF member.

The SERIALIZE statement does not affect the step being targeted to a specific image (with a TARGET JTL statement or a TARGET statement in the UCF member).

Specifying the NOSERIALIZE Statement

Figure 7-10 shows the format for the NOSERIALIZE statement.

Figure 7-10 BSLCNTL NOSERIALIZE



You can specify NOSERIALIZE at the step level. The NOSERIALIZE statement instructs Job Optimizer to allow the step to run in parallel with other steps.

Typically, you use SERIALIZE in the UCF member to force the steps associated with a program (PROGRAM=) or a DD statement (DDNAME=) to run serially (not concurrently with other steps). To override the global nature of the UCF member, include the NOSERIALIZE JTL statement at the step level (after the EXEC PGM= statement) of a job to instruct Job Optimizer to allow the step to run in parallel with other steps.

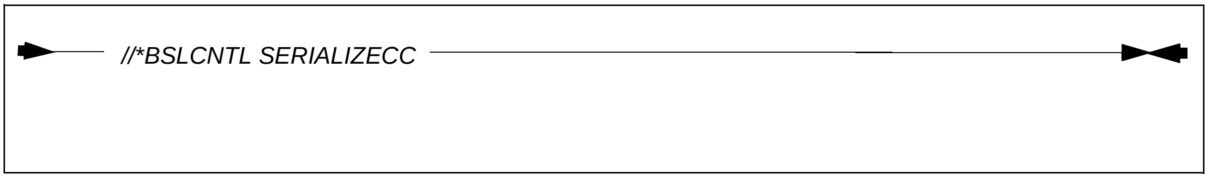
Because JTL statements override UCF statements, you can use the NOSERIALIZE JTL statement to override a SERIALIZE statement in the UCF member.

The NOSERIALIZE statement does not affect the step being targeted to a specific image (with a TARGET JTL statement or a TARGET statement in the UCF member).

Specifying the SERIALIZECC Statement

Figure 7-11 shows the format for the SERIALIZECC statement.

Figure 7-11 BSLCNTL SERIALIZECC

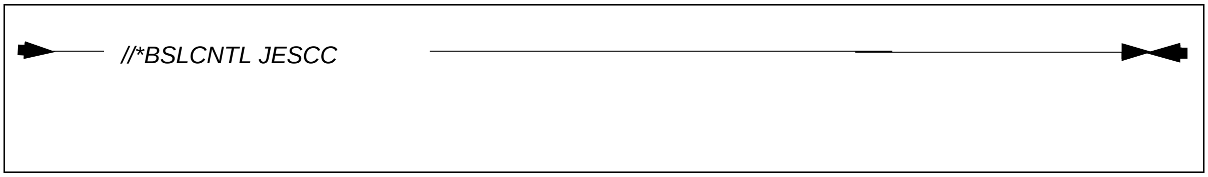


SERIALIZECC is a combination of SERIALIZE and JESCC actions. The SERIALIZECC statement specifies that this step is to be serialized in relationship to other steps. That is, all steps prior to this step must complete prior to this step beginning, and no subsequent steps may begin execution until after this step is completed. The step will run in the JES initiator, and condition codes on the EXEC statement will be honored. This option will override the job BSLCNTL NOCC option.

Specifying the JESCC Statement

Figure 7-12 shows the format for the JESCC statement.

Figure 7-12 BSLCNTL JESCC

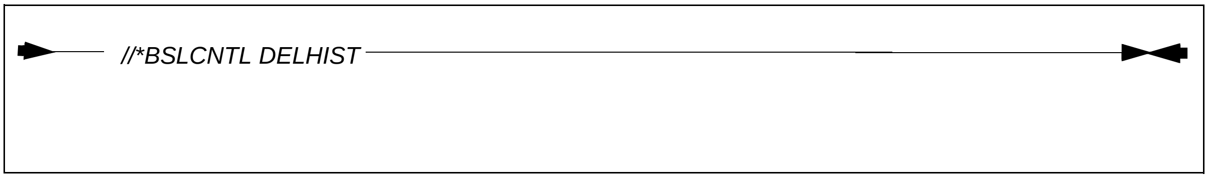


The step will run in the JES initiator, and condition codes on the EXEC statement will be honored. This option will override the job BSLCNTL NOCC option.

Specifying the DELHIST Statement

Figure 7-13 shows the format for the DELHIST statement.

Figure 7-13 BSLCNTL DELHIST

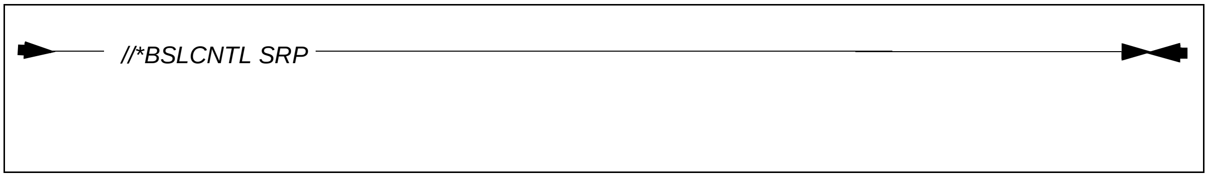


You can specify DELHIST at the job level only. Step-level and DD-level statements never override a job-level DELHIST statement. When you specify DELHIST, all previous history will be deleted, and the job will be analyzed.

Specifying the SRP Statement

Figure 7-14 shows the format for the Shared Record Positioning (SRP) statement.

Figure 7-14 BSLCNTL SRP



You can specify SRP at the following levels:

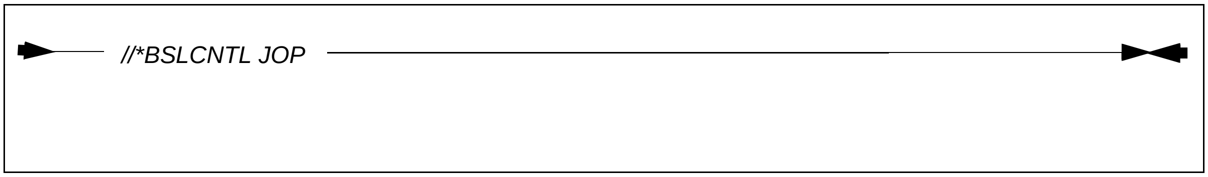
- job level by placing the statement before the first EXEC statement in the job
- step-level by placing the statement after the EXEC statement in the job

When you include an SRP statement, Job Optimizer will use SRP as the piping transport.

Specifying the JOP Statement

Figure 7-15 shows the format for the Job Optimizer Pipes (JOP) statement.

Figure 7-15 BSLCNTL JOP



You can specify JOP at the following level:

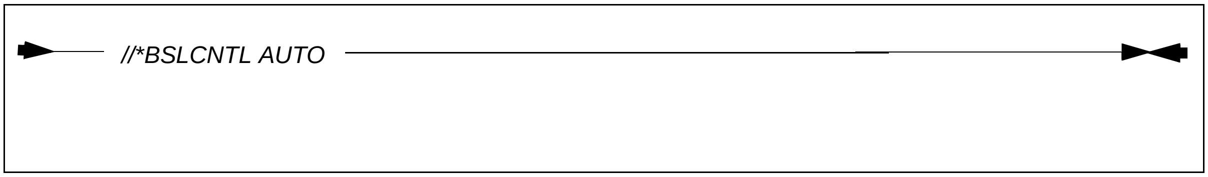
- job level by placing the statement before the first EXEC statement in the job

When you include a JOP statement, Job Optimizer will use JOP as the piping transport.

Specifying the AUTO Statement

Figure 7-16 shows the format for the AUTO statement.

Figure 7-16 BSLCNTL AUTO



You can specify AUTO at the following level:

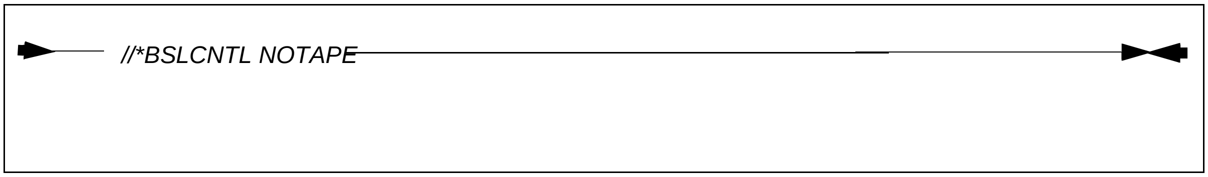
- job level by placing the statement before the first EXEC statement in the job

When you include a AUTO statement, Job Optimizer chooses the transport type.

Specifying the NOTAPE Statement

Figure 7-17 shows the format for the NOTAPE statement.

Figure 7-17 BSLCNTL NOTAPE

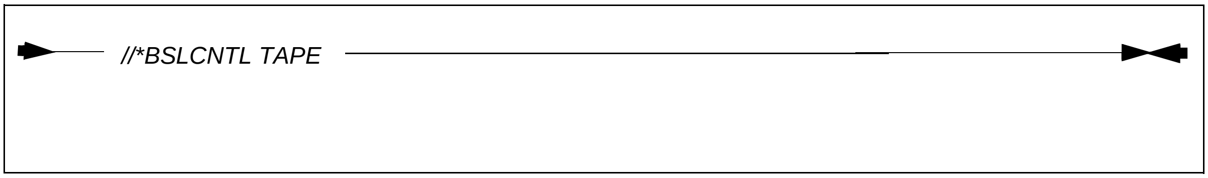


You can specify NOTAPE only at the job level. The step or steps within this job that utilize tape data sets will not be eligible to be split regardless of the specification for “TAPESPLIT” in the Job Performance Options.

Specifying the TAPE Statement

Figure 7-18 shows the format for the TAPE statement.

Figure 7-18 BSLCNTL TAPE



You can specify TAPE only at the job level. The step or steps within this job that utilize tape data sets are eligible to be split regardless of the specification for “TAPESPLIT” in the Job Performance Options.

Chapter 8 Job Optimizer Pipes Introduction

This chapter describes Job Optimizer Pipes, its functions, availability, and components. This chapter discusses the following topics:

Overview

Job Optimizer Pipes provides in-memory piping of application data between batch jobs and job steps. By using pipes, data-related applications can execute concurrently rather than sequentially, reducing the elapsed time that is required to process the jobs.

Jobs and job steps generally run in sequence because they depend on data files that become available only after the application that creates them completes execution. This problem is compounded by the fact that large sequential files are written/read to/from DASD or tape. Heavy sequential I/O (input/output) processing results in I/O delays and decreased CPU usage. This inefficiency is unnecessary, and Job Optimizer Pipes can eliminate it.

Job Optimizer Pipes significantly reduces I/O operations and enables replacing sequential processing with parallel processing wherever feasible. Through parallel processing, Job Optimizer Pipes increases batch processing efficiency.

Job Optimizer Pipes is highly effective in single-system and multi-system Parallel Sysplex environments. Job Optimizer Pipes efficiently applies the parallel processing capabilities of the Parallel Sysplex environment to batch processing.

Job Optimizer Pipes Functions

Job Optimizer Pipes provides the following capabilities:

- pipes

  Data is transferred between applications that run in parallel without using external storage. Pipes are discussed in detail in the following pages.

- dynamic pipe setting

  No JCL changes are required. JCL references to physical files are dynamically replaced by references to pipes.

- physical file creation

  Data transferred using a pipe can also be written to a physical file if a physical copy of the data is required for future use or for backup purposes.

- Applications internal checkpoint support

Internal checkpointing is supported when using pipes instead of sequential datasets.

Pipes

A pipe is a processor storage buffer that enables data to be passed between applications without using DASD or tape.

Moving data using processor storage is quicker than writing it to or reading it from external storage. This is an important advantage of using pipes.

But the main advantage of using pipes is that it enables true parallel processing. An application does not need to finish running before the data it generates is available to another application. This advantage can be easily understood by the following analogy.

- Parallel Processing: Bottling Beverages in an Assembly Line

  As a bottle moves along the assembly line in a beverage bottling plant, one machine fills the bottle and another machine caps the bottle, etc.

  The assembly line is built for efficiency. All machines work simultaneously (in parallel), each machine acting upon a different bottle. No machines sit idle.

  One bottle is being filled while a previous one is being capped and another bottle is being labeled while a previous one is being placed in a carton.

  By the time the last few bottles are being filled, almost all bottles have been completely processed and packaged.

- Sequential Processing: Bottling Beverages One Machine at a Time

  Suppose, however, that instead of using an assembly line (meaning, parallel processing), the bottling processes were performed sequentially (meaning, one process at a time). Crating of the bottles could not begin until all bottles were labeled. But labeling could not begin until all bottles were capped. And capping could not begin until all bottles were filled. The inefficiency and wasted time in such a process is readily apparent. Most machines would be idle most of the time. A bottling plant would never run this way. Yet, this is how many sites normally process data in the production environment.

Data Flow between Applications

This section describes data flow between applications, both without pipes and with pipes showing how pipes can “turn your data processing operation into an assembly line”.

Data Flow without Pipes

An application is a set of steps to be processed in a sequence. Typically, an application processes sequential files as follows:

1. Required datasets are opened and, data is read.
2. Data is processed sequentially.
3. Processed data is written to other datasets.
4. When all data has been processed, the datasets are closed. The written data is then available for other applications.

When an application is running, the datasets it creates are locked and cannot be accessed by another (successor) application. Successor applications cannot begin until the predecessor application has completed.

As shown in Table 8-1, a sequential dataset is used to transfer data between the applications. All data items are processed by application APPL1 before application APPL2 begins processing. In this example, eight units of time are required to process four data items.

Table 8-1 Sequential Processing (Part 1 of 2)

Elapsed Time Units	Data Processing
	APPL1 (Predecessor)
	Open Dataset A (For Output)
1	Process & Write Data-Item1
2	Process & Write Data-Item2
3	Process & Write Data-Item3
4	Process & Write Data-Item4
	Close Dataset A
	APPL2 (Successor)
	Open Dataset A (For Input)
5	Read & Process Data-Item1
6	Read & Process Data-Item2
7	Read & Process Data-Item3

BMC Software, Inc., Confidential and Proprietary Information

Table 8-1 Sequential Processing (Part 2 of 2)

Elapsed Time Units	Data Processing
8	Read & Process Data-Item4
	Close Dataset A

As Table 8-1 on page 8-4 shows, predecessor and successor applications process 108,000 records at the rate of 36,000 per hour (10 per second). The predecessor application must execute for 3 hours before the successor application can begin. The total processing time for both applications is about 6 hours.

Data Flow with Pipes

Just as water flows through pipes from a source (for example, a reservoir) to a destination (for example, a shower), data can flow through pipes from a source (predecessor application) to a destination (successor application).

When pipes are used:

- After a unit of data is processed by a predecessor application, it is available to be processed by the successor application.
- While the successor application processes this unit of data, the predecessor application can process the next unit of data (in parallel).

As shown in Table 8-2, a pipe is used to transfer data instead of a sequential dataset. In Table 8-2, only five units of time are required to process four data items because the applications run in parallel.

Table 8-2 Pipe Processing

Elapsed Time Units	APPL1 (Predecessor)	Data in Pipe	APPL2 (Successor)
	Open Dataset A (For Output)		Open Dataset A (For Input)
1	Process & Write Data-Item1 ⇒	⇒ Data-Item1	
2	Process & Write Data-Item2 ⇒	Data-Item1 ⇒ ⇒ Data-Item2	⇒ Read & Process Data-Item1
3	Process & Write Data-Item3 ⇒	Data-Item2 ⇒ ⇒ Data-Item3	⇒ Read & Process Data-Item2
4	Process & Write Data-Item4 ⇒	Data-Item3 ⇒ ⇒ Data-Item4	⇒ Read & Process Data-Item3
5	Close Dataset A	Data-Item4 ⇒	⇒ Read & Process Data-Item4
			Close Dataset A

As in Table 8-1 on page 8-4, both the predecessor and successor applications in Table 8-2 process 108,000 records at the rate of 36,000 records per hour (10 per second). The total processing time (discounting minor differences) is just over 3 hours, about half the time that is required for sequential processing.

Pipe/Dataset Comparison

The pipe can be considered a virtual dataset in storage. Table 8-3 compares a pipe and a dataset.

Table 8-3 Comparison of a Pipe and a Dataset

Pipe	Dataset
resides in processor storage	resides on external storage
assigned a dataset name	assigned a dataset name
holds a predefined number of data items	holds all data items
an unlimited amount of data can pass through a pipe	the amount of data the dataset can hold (File size) is determined by allocation
data is normally not retained after processing; a Job Optimizer Pipes option enables data to be written to a dataset	data is retained in the dataset after processing (according to the dataset disposition definition).

Advantages of Using Pipes

The main advantage of passing data through processor storage (using pipes) instead of sequential files is that this method enables parallel processing of applications.

The effect that parallel processing can have on processing time is enormous:

- When processing applications sequentially, total processing time approximates the sum of the processing times of each application.

- When processing applications in parallel, total processing time approximates the processing time of the longest running application. The greater the number of applications using the same data, the greater the savings from parallel (versus sequential) processing.

In addition, passing data through processor storage:

- Reduces the number of physical I/O operations.
- Reduces contention for I/O resources.

- Reduces tape and DASD usage.
- Increases CPU utilization.

Pipe Terminology

Pipe processing always involves the components that are described in Table 8-4.

Table 8-4 Pipe Components

Component	Description
Pipe	a processor storage buffer that can hold a specified number of data items
Writer	an application that writes data to a pipe
Reader	an application that reads data from a pipe

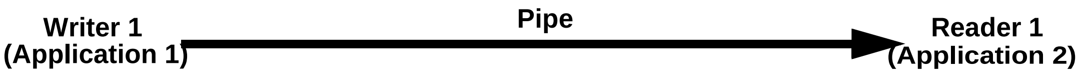
In pipe terminology, the writers and readers of a pipe are called participants.

To meet the needs of the production environment, multiple writers, readers and pipes can be interconnected. The combination of interconnected pipes, writers and readers is called a collection.

Simple Collection

The simplest collection, as shown in Figure 8-1, consists of a single writer, a single reader and a single pipe.

Figure 8-1 Single Writer, Single Reader, and Single Pipe Example



Multiple Readers/Writers Collections

Multiple writers/readers can be connected to a single pipe. This is shown in Figure 8-2, Figure 8-3, and Figure 8-4.

Figure 8-2 Single Writer, Multiple Readers, and Single Pipe Example

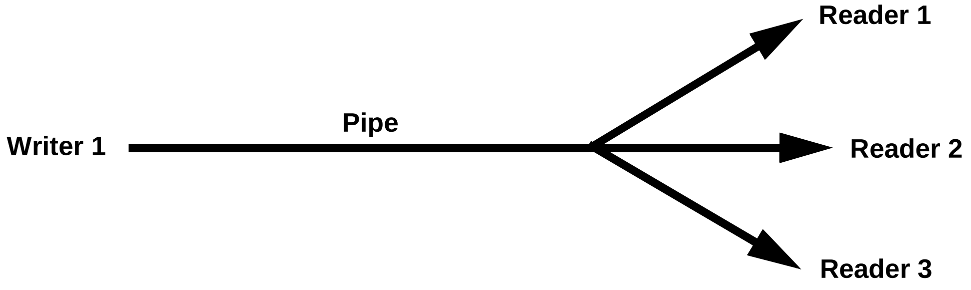


Figure 8-3 Multiple Writers, Single Reader, and Single Pipe Example

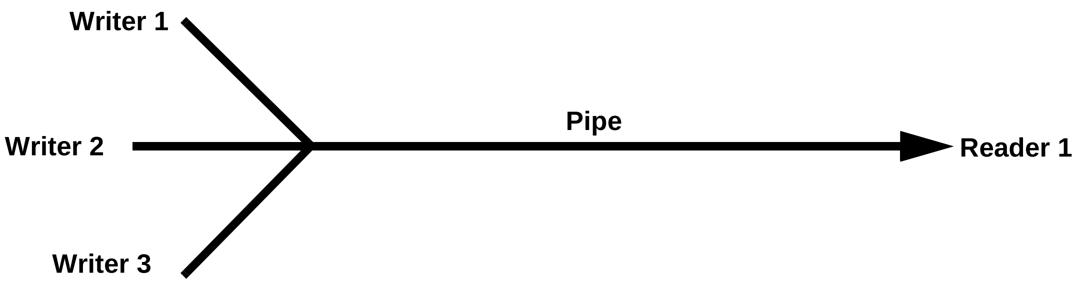
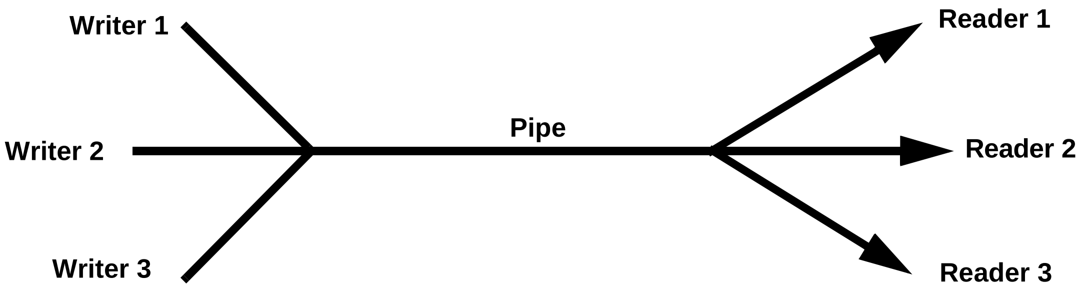


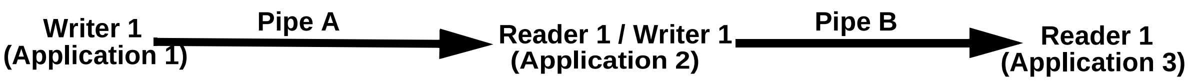
Figure 8-4 Multiple Writers, Multiple Readers, and Single Pipe Example



Multiple Pipes Collection

Multiple pipes can be defined in a collection. Each pipe has its own participants (writers and readers) and an application can act as a reader from one pipe and a writer to another pipe, as shown in Figure 8-5 and Figure 8-6 on page 8-9. The writers/readers are numbered relative to the other writers/readers of the same pipe.

Figure 8-5 Single Writer, Single Reader, and Multiple Pipes Example



In Figure 8-5, there are two pipes (Pipe A and Pipe B) and three applications are using these pipes. Pipe A has two participants (one writer and one reader):

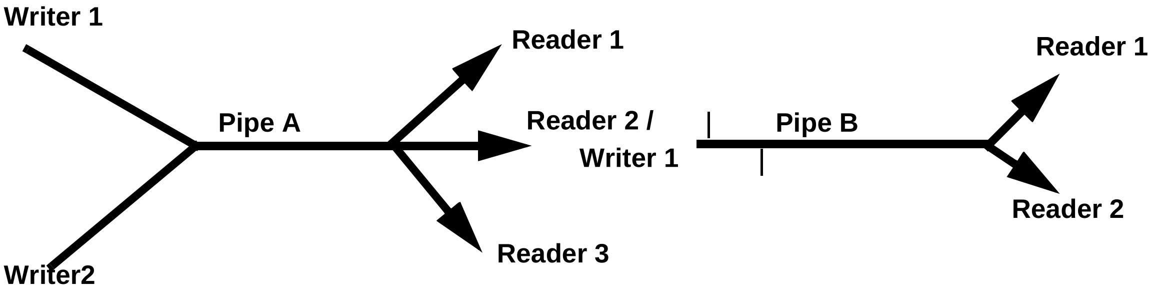
- Writer 1 (Application 1)
- Reader 1 (Application 2)

Pipe B has two participants (one writer and one reader):

- Writer 1 (Application 2)
- Reader 1 (Application 3)

Application 2 is a reader of Pipe A and a writer of Pipe B.

Figure 8-6 Multiple Writers, Multiple Readers, and Multiple Pipes Example



In Figure 8-6, there are two pipes (Pipe A and Pipe B) and seven applications that are using these pipes. Pipe A has five participants (two writers and three readers). Pipe B has three participants (one writer and two readers). The second reader of Pipe A is the writer of Pipe B.

BMC Software, Inc., Confidential and Proprietary Information

Supported Pipe Types

Job Optimizer Pipes supports the following types of pipes:

- job-to-job pipes
- step-to-step pipes
- BatchPipes-compatible pipes

Job-to-Job Pipes

Job-to-job pipes are pipes used between batch jobs, allowing the jobs to run in parallel. Job-to-job pipes can be dynamically set by Job Optimizer Pipes. This allows the original job’s JCL to remain unchanged. The job is eligible to run sequentially (without pipes) or run in parallel (using pipes). Job-to-job pipes are managed by using Pipe Policy Rules. These rules also define which datasets are to be replaced by pipes.

Step-to-Step Pipes

Step-to-step pipes allow data transfer between steps. The Job Optimizer component of MAINVIEW Batch Optimizer uses step-to-step piping to allow data-dependent steps to execute in parallel. Job Optimizer dynamically sets a step-to-step pipe in the steps running in parallel, allowing the reader step to access a block as soon as it is written by the writer step. No user definitions are required.

BatchPipes-Compatible Pipes

Job Optimizer Pipes supports BatchPipes SUBSYS= subsystem parameters. When converting from BatchPipes (or SmartBatch) to Job Optimizer Pipes, the jobs' JCL can remain unchanged. Job Optimizer Pipes analyzes the BatchPipes subsystem parameters and defines and manages pipes accordingly. See Appendix E, “Migrating from BatchPipes to Job Optimizer Pipes.” for a complete description of BatchPipes-Compatible Pipes.

Single-System and Multi-System Support

When working in a single system or in a multi-system non-Parallel Sysplex environment, Job Optimizer Pipes works in Local mode. In this mode, each system is managed separately. Applications running in parallel and sharing a pipe cannot run in different systems.

When working in a multi-system Parallel Sysplex environment, Job Optimizer Pipes can work in Local mode or Global mode. When working in Local mode, Job Optimizer Pipes manages each system separately. When working in Global mode, all the systems are handled as one system, allowing applications running in parallel and sharing a pipe to run in different systems.

When all participants of a pipe run in the same system, the pipe is called a Local pipe. (Local Pipes can be used when working in Local mode or Global mode.) When the participants of a pipe run in different systems, the pipe is called a Global pipe. (Global pipes can be used only when working in Global mode.)

A Global pipe is managed by Job Optimizer Pipes in the system where the first writer of the pipe is executing. This system is called the Pipe Home System. Participants of a Global pipe that run in the Pipe Home System are called Local participants. Participants of a Global pipe that do not run in the Pipe Home System are called Remote participants.

Job Optimizer Pipes Components

Job Optimizer Pipes comprises the following components:

- Pipe Policy Rules
- Job Optimizer Pipes address space
- Job Optimizer Pipes data space
- Job Optimizer Pipes subsystem

BMC Software, Inc., Confidential and Proprietary Information

Pipe Policy Rules

Pipe Policy Rules (also called Pipe Rules) are user-defined parameters that determine how Job Optimizer Pipes will set and manage job-to-job pipes and BatchPipes-compatible pipes (when BatchPipes phased migration is used). The Pipe Policy Rules are identified by their dataset name. For job-to-job pipes they contain definitions that are used by Job Optimizer Pipes for pipe control (for example, number of writers and readers) and selection criteria to identify candidates for pipe processing. For BatchPipes-compatible pipes they only identify candidates for using Job Optimizer Pipes services instead of BatchPipes services.

The Pipe Policy Rules are saved in partitioned data set members (called Pipe Policy Tables).

Pipe Rule Defaults: JOPDEFRL

JOPDEFRL is the Pipe Rule Defaults member. It contains default values for all fields in the rule definition. When a job-to-job pipe is used, Job Optimizer Pipes automatically assigns values from the Pipe Rule Defaults for fields that are not filled in by the user during Pipe Policy rule definition.

The values in the Pipe Rule Defaults can be site-customized.

Job Optimizer Pipes Address Space

The Job Optimizer Pipes address space is a started task that runs in each system in which Job Optimizer Pipes is operational. The address space handles rules, pipe services and buffers and user services such as operator commands. The address space’s main roles are rules management, pipes initialization/termination, and pipes “housekeeping.”

Job Optimizer Pipes Data Space

The Job Optimizer Pipes Data Space is used to hold pipe buffers, pipe control blocks, and so on. It is managed by the Job Optimizer Pipes address space and is accessed by the Job Optimizer Pipes address space and by Job Optimizer Pipes subsystem.

Job Optimizer Pipes Subsystem

The Job Optimizer Pipes subsystem has two functions:

- The Real-Time Job Handler performs dynamic pipe setting and BatchPipes Phased Migration setting.
- The Pipe Handler manages the pipes and emulates standard sequential data set process to the application program.

Real-Time Job Handler

The Real-Time Job Handler is activated when a job begins execution in the system. It examines the pipe rules to determine whether any of the job's datasets are candidates for pipe processing. If job-to-job piping is required, the Real-Time Job Handler replaces references to the dataset with references to the Job Optimizer Pipes subsystem. If BatchPipes Phased Migration is required, the Real-Time Job Handler replaces references to the BatchPipes subsystem with references to the Job Optimizer Pipes subsystem.

Pipe Handler

The Pipe Handler is activated for each pipe management or I/O operation done by the application to the DD statement referencing a pipe (explicitly defined or dynamically set by the Real Time Job Handler). It manages all pipe activity (for example, pipe definition, pipe initialization, and pipe I/O) by using the services of the Job Optimizer Pipes address space. This process is transparent to the application program. The application continues using standard sequential dataset processing.

Jobs Parallel Processing and Scheduling

Efficient parallel processing requires that executing applications do not sit idle, waiting for external conditions to be satisfied, before they continue processing. Otherwise, the following problems could occur:

- If a reader application begins execution before its predecessor writer application, it will have nothing to read (it will be connected to an empty pipe).

- If a writer application is executing and its successor reader application is not executing shortly thereafter, the pipe connected to the writer application will fill with data and the writer application will be unable to continue processing.

CONTROL-M users can use the special interface with Job Optimizer Pipes to achieve efficient parallel processing. This interface is described in Appendix F, “IOA and INCONTROL User Considerations.”

Chapter 9 Job Optimizer Pipes Dialog

This chapter describes Job Optimizer Pipes Pipe Policy List, Pipe Policy Tables, and Pipe Policy Rules, also called Pipe Rules. This chapter includes information about the panels and dialog actions that you use to view and modify these entities. This chapter discusses the following topics.

BMC Software, Inc., Confidential and Proprietary Information

BMC Software, Inc., Confidential and Proprietary Information

Overview

This section describes Pipe Policy Lists, Pipe Policy Tables, and Pipe Policy Rules.

Understanding Pipe Policy List

The Pipe Policy List defines a list of Pipe Policy Tables to be activated (loaded) during Job Optimizer Pipes initialization. Each Pipe Policy Table contains one or more rules that define data set candidates for Job Optimizer Pipes processing.

The Pipe Policy List is kept in a partitioned dataset member named PIPLST*xx*, residing in the MAINVIEW Batch Optimizer control data set. The Pipe Policy List is identified by the BatchPlex member PIPELIST parameter.

Understanding Pipe Policy Table

A Pipe Policy Table contains a set of pipe policy rules. Each rule consists of a group of records that define data set candidates for Job Optimizer Pipes processing. They identify operational characteristics of the pipe as well as the data set and job selection criteria. Each rule consists of the following information:

- rule identification (information such as the rule name and the pipe data set name)
- pipe attributes (data set attribute information)
- synchronization (how to synchronize pipe participants)
- maximum wait time (how long pipe participants wait for certain pipe condition milestones)
- writer information regarding the writer(s) of the pipe (data set)
- reader information regarding the reader(s) of the pipe (data set)

Pipe Policy Tables are kept in a partitioned dataset member named PIPPOL*xx*. They can reside in any partitioned data set that Job Optimizer Pipes is authorized to read from. The Pipe Policy List (identified by the BatchPlex member PIPELIST parameter) determines which Pipe Policy Tables to activate during Job Optimizer Pipes initialization. After initialization, however, you can use the modify command or the MAINVIEW Batch Optimizer dialog to activate other Pipe Policy Tables or replace an active Pipe Policy Table with an updated version.

Registering a Pipe Policy List

The Pipe Policy List defines a list of Pipe Policy Tables to be activated (loaded) during Job Optimizer Pipes initialization. This section describes how to use the MAINVIEW Batch Optimizer dialog to perform Pipe Policy List registration tasks.

MAINVIEW Batch Optimizer includes online Help for each panel. It also provides input field help to describe operational considerations and valid field values. For more information on accessing the MAINVIEW Batch Optimizer online Help, see Appendix B, “Navigating the User Interface.”

You created a sample Pipe Policy List during the customization process in preparation to execute the installation verification jobs. You can tailor the set of rules that are loaded during Job Optimizer Pipes initialization using one or more of the following methods:

- add new Pipe Policy Tables to the Pipe Policy List

  In these tables you include additional rules.

- modify the sample Pipe Policy Table to include additional rules

  For more information, see “Registering a Pipe Policy List” on page 9-4.

- Create a new Pipe Policy List that will include all the required Pipe Policy Tables.

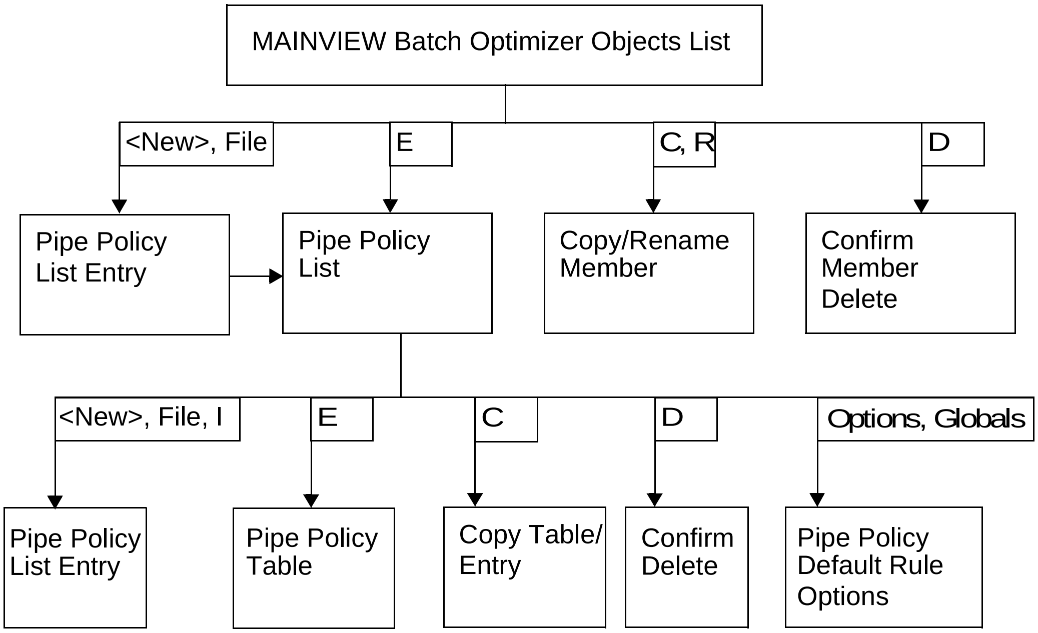
  If this method is chosen, BatchPlex member PIPELIST parameter has to be modified to identify the new Pipe Policy List.

Panel Flow for a Pipe Policy List Member

Figure 9-1 shows the panel flow when you are using the MAINVIEW Batch Optimizer dialog to edit a Pipe Policy List. This is the basic panel flow that indicates the action codes (E, I, C, R, D), commands (New, Globals), or the pull-down option (File, Options) that you select to move from one panel to another.

Note: Editing or viewing a Pipe Policy List is explained in this chapter. For more information about creating a new Pipe Policy List, see Appendix B, "Navigating the User Interface."

Figure 9-1 Pipe Policy List Registration–Dialog Panel Flow



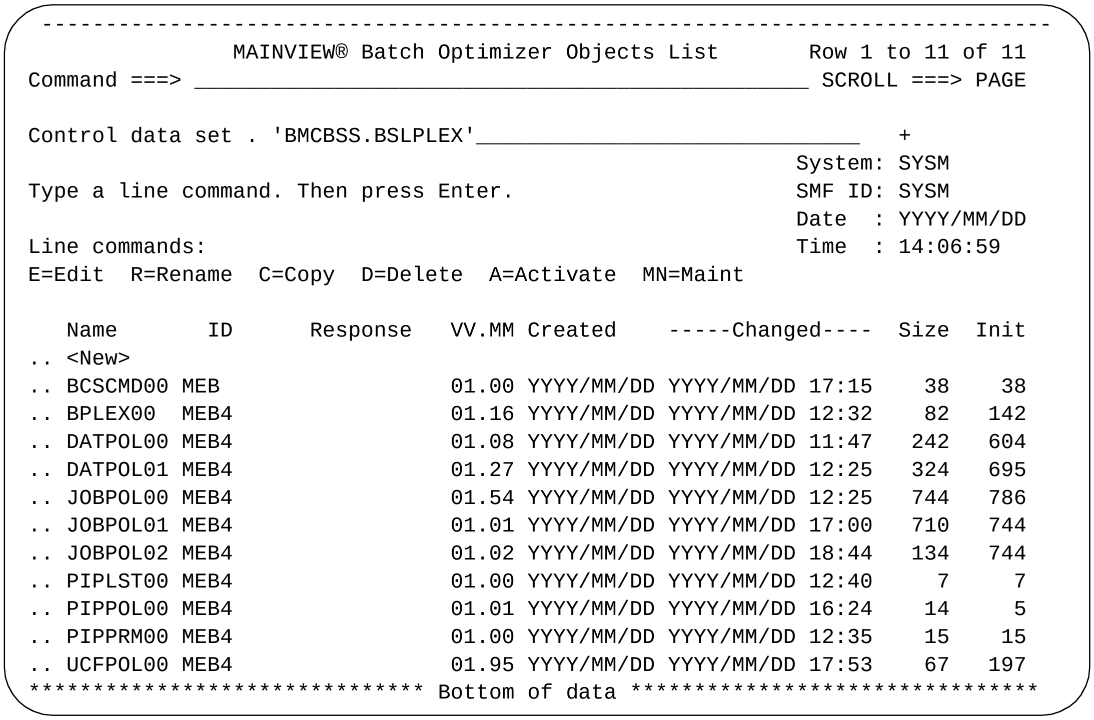
Editing or Viewing a Pipe Policy List

You can edit or view the information in a Pipe Policy List.

To edit or view information for a Pipe Policy List in the control data set, complete the following steps:

Step 1 Access the MAINVIEW Batch Optimizer Objects List panel, shown in Figure 9-2.

Figure 9-2 MAINVIEW Batch Optimizer Objects List Panel

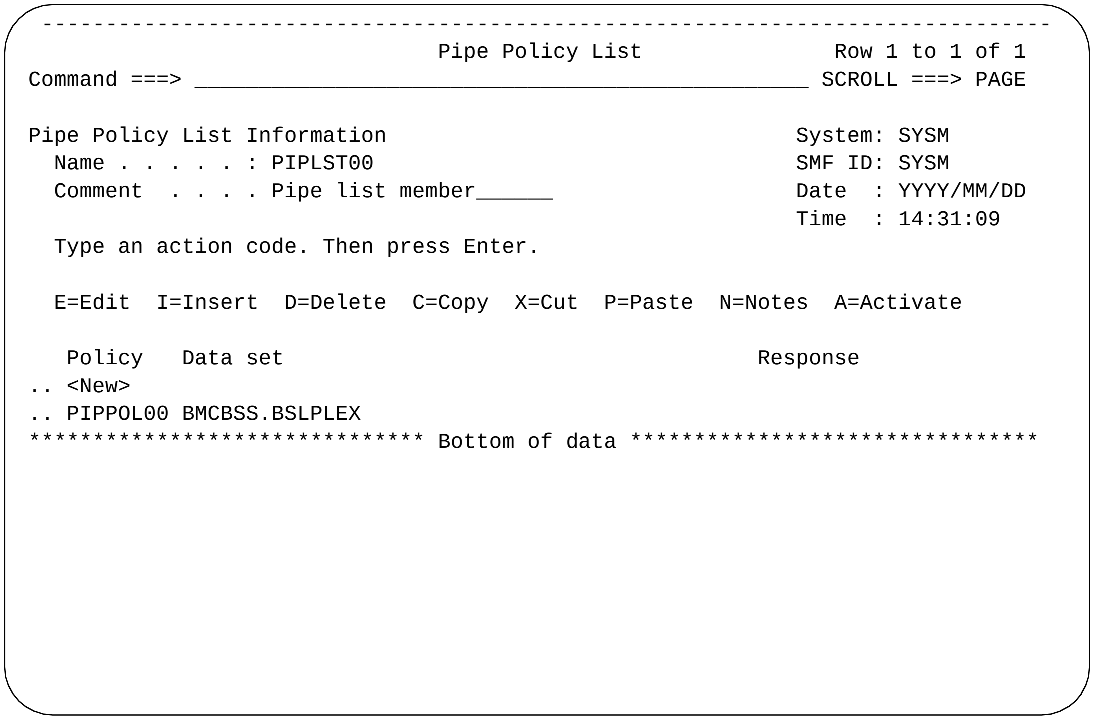


```
 ------------------------------------------------------------------------------
                MAINVIEW® Batch Optimizer Objects List       Row 1 to 11 of 11
Command ===> ________________________________________________ SCROLL ===> PAGE

Control data set . 'BMCBSS.BSLPLEX'______________________________   +
                                                            System: SYSM
Type a line command. Then press Enter.                      SMF ID: SYSM
                                                            Date  : YYYY/MM/DD
Line commands:                                              Time  : 14:06:59
E=Edit  R=Rename  C=Copy  D=Delete  A=Activate  MN=Maint

   Name       ID      Response   VV.MM Created    -----Changed----  Size  Init
.. <New>
.. BCSCMD00 MEB                  01.00 YYYY/MM/DD YYYY/MM/DD 17:15    38    38
.. BPLEX00  MEB4                 01.16 YYYY/MM/DD YYYY/MM/DD 12:32    82   142
.. DATPOL00 MEB4                 01.08 YYYY/MM/DD YYYY/MM/DD 11:47   242   604
.. DATPOL01 MEB4                 01.27 YYYY/MM/DD YYYY/MM/DD 12:25   324   695
.. JOBPOL00 MEB4                 01.54 YYYY/MM/DD YYYY/MM/DD 12:25   744   786
.. JOBPOL01 MEB4                 01.01 YYYY/MM/DD YYYY/MM/DD 17:00   710   744
.. JOBPOL02 MEB4                 01.02 YYYY/MM/DD YYYY/MM/DD 18:44   134   744
.. PIPLST00 MEB4                 01.00 YYYY/MM/DD YYYY/MM/DD 12:40     7     7
.. PIPPOL00 MEB4                 01.01 YYYY/MM/DD YYYY/MM/DD 16:24    14     5
.. PIPPRM00 MEB4                 01.00 YYYY/MM/DD YYYY/MM/DD 12:35    15    15
.. UCFPOL00 MEB4                 01.95 YYYY/MM/DD YYYY/MM/DD 17:53    67   197
******************************* Bottom of data ********************************
```

Step 2 Position the cursor in the action entry field to the left of your choice of Pipe Policy List. Type **E** (Edit), and press **Enter**.

The Pipe Policy List panel is displayed and shows the Pipe Policy List Entries. Each entry points to a Pipe Policy table that is defined in this list. Figure 9-3 on page 9-7 shows an example of the Pipe Policy List panel, listing a single Pipe Policy List entry.

Figure 9-3 Pipe Policy List Panel



```
 -------------------------------------------------------------------------------
                                Pipe Policy List               Row 1 to 1 of 1
Command ===> ________________________________________________ SCROLL ===> PAGE

Pipe Policy List Information                                System: SYSM
  Name . . . . . : PIPLST00                                 SMF ID: SYSM
  Comment  . . . . Pipe list member______                   Date  : YYYY/MM/DD
                                                            Time  : 14:31:09
  Type an action code. Then press Enter.

  E=Edit  I=Insert  D=Delete  C=Copy  X=Cut  P=Paste  N=Notes  A=Activate

   Policy   Data set                                     Response
.. <New>
.. PIPPOL00 BMCBSS.BSLPLEX
******************************* Bottom of data ********************************
```

Step 3 To make changes to any entry field listed on this panel, position the cursor to the field, clear the field, then type in the new text.

Step 4 You can make changes to the Pipe Policy Tables defined in the Pipe Policy List panel. For more information, see “Editing or Viewing a Pipe Policy Table” on page 9-16.

Step 5 You can make changes to the Pipe Policy Rule defaults. For more information, see “Editing or Viewing Pipe Rule Defaults” on page 9-18.

Modifying Pipe Policy List Entries

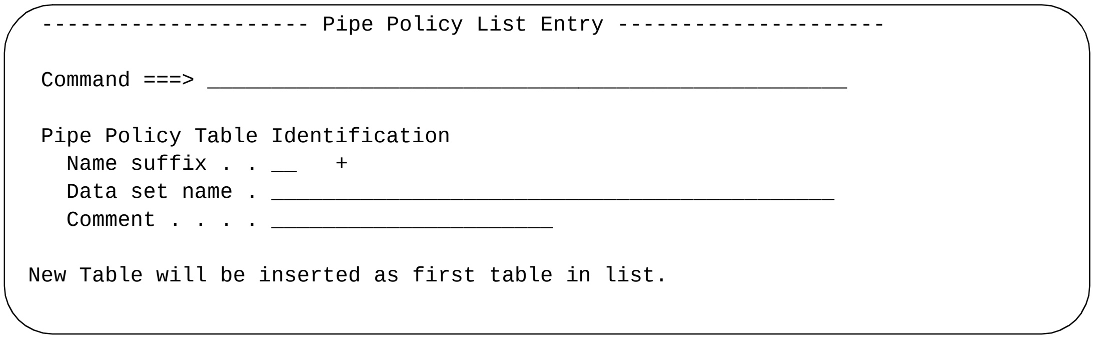
You can modify the Pipe Policy List entries and the Pipe Policy Tables that are pointed to by the entries. The order of the entries in the Pipe Policy List is not important. When Job Optimizer Pipes processes the Pipe Policy List, Job Optimizer Pipes sorts the rules according to the Best Match Order. For more information, see Chapter 12, “Job Optimizer Pipes Implementation Considerations.”

To modify information in a Pipe Policy List, complete the following steps:

Step 1 Perform the following administrative tasks to make changes to the Pipe Policy list entries:

- To add a new entry to the top of the list, select 1 (New) from the File pull-down or type E (Edit) in the input field left of list entry <New>, and press **Enter**. The Pipe Policy List Entry panel, shown in Figure 9-4, is displayed, indicating the position of the new entry in the list. Edit the new definition.
- To insert (similar to add, but in your choice of list position) a new entry, decide which list entry you want the new entry to be placed after. Position the cursor in the input field left of your choice of list entry. Type **I** (Insert) and press **Enter**. The Pipe Policy List Entry panel shown in Figure 9-4 is displayed, indicating the position of the new entry in the list. Edit the new definition.

Figure 9-4 Pipe Policy List Entry Panel



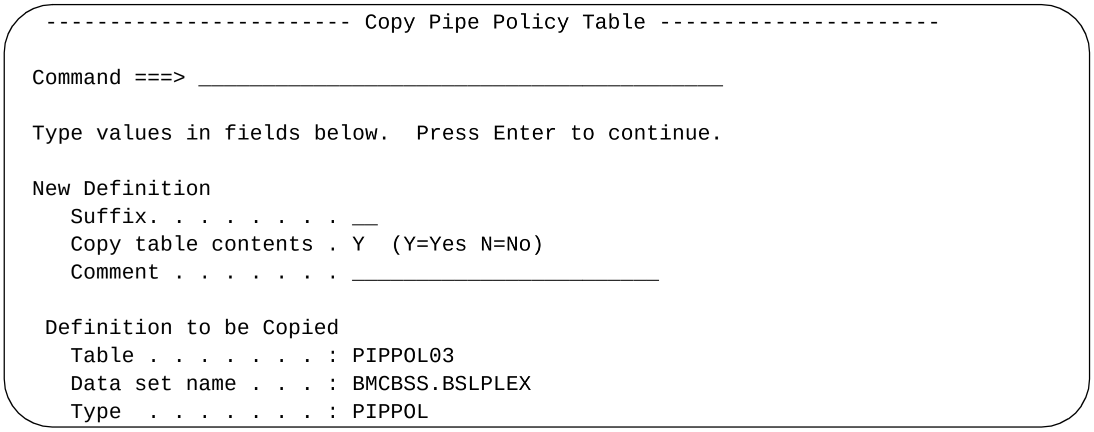
```
 -------------------- Pipe Policy List Entry --------------------

 Command ===> __________________________________________________

 Pipe Policy Table Identification
   Name suffix . . __   +
   Data set name . ____________________________________________
   Comment . . . . ______________________

New Table will be inserted as first table in list.
```

The Name suffix field supports the Prompt command. The inserted or added Pipe Policy List entry may be a new or pre-existing Pipe Policy Table. Press **Enter** on the Name suffix field to obtain a list of Pipe Policy Tables residing in the control data set.

Note: If a new Pipe Policy Table name is added to the list, a new Pipe Policy Table is created in the control data set. This Pipe Policy Table will be saved when the Pipe Policy List is saved.

- To copy (and paste) an entry and the Pipe Policy Table (optional), position the cursor in the input field to the left of your choice of entry. Type **C** (Copy), and press **Enter**. The Copy Pipe Policy Table panel is displayed, as shown in Figure 9-5. Specify a target member name suffix for the copy operation and whether to copy the table contents or not, and press **Enter**. The message “Definition (selected entry) copied to clipboard as (new entry)” is displayed. To choose a destination for the copied entry, type **P** (Paste) to the left of the line after which you want to place the copied entry and press **Enter**. The definition is moved from the clipboard to the list at the selected position. If the table contents were copied, a new table is created with the contents of the copied table. This table will be saved when the Pipe Policy List is saved.

Figure 9-5 Copy Pipe Policy Table Panel



```
 ----------------------- Copy Pipe Policy Table ---------------------

Command ===> _________________________________________

Type values in fields below.  Press Enter to continue.

New Definition
   Suffix. . . . . . . . __
   Copy table contents . Y  (Y=Yes N=No)
   Comment . . . . . . . ________________________

 Definition to be Copied
   Table . . . . . . . : PIPPOL03
   Data set name . . . : BMCBSS.BSLPLEX
   Type  . . . . . . . : PIPPOL
```

- To move (cut and paste) an entry, position the cursor in the input field, to the left of your choice of entry. Type **X** (Cut), and press **Enter**. If the clipboard is empty, the definition is moved from the list to the clipboard and responds with “Definition cut to clipboard and deleted from definition list.” To choose a destination for the cut entry, type **P** (Paste) to the left of the line following which you want to place the cut entry and press **Enter**. The definition is moved from the clipboard to the list at the selected position.

- To delete an entry, position the cursor in the input field to the left of your choice of entry. Type **D** (Delete), and press **Enter**. The Confirm Delete pop-up panel is displayed, which lets you confirm the delete request before removing the entry from the pipe policy list.

Step 2 You can save your changes by using one of the following methods:

- Select the File pull-down choice 2 (Save Pipe Policy List member to disk). The changed Pipe Policy List, the Pipe Rule defaults, and the Pipe Policy Tables that are defined in the Pipe Policy List are saved. When the save is complete, the Pipe Policy List panel is displayed.

- Press **F3** (Exit). The Confirm Exit pop-up panel is displayed. Type **1** (Save changes to disk and exit), and press **Enter**. The changed Pipe Policy List and any copied or pasted Pipe Policy Tables are saved. Any changes that have been performed to the Pipe Rule defaults are also saved. When the save is complete, the MAINVIEW Batch Optimizer Objects List panel is displayed.

Activating a Pipe Policy List

BatchPlex member PIPELIST parameter defines the current Pipe Policy List. When Job Optimizer Pipes is started, the Pipe Policy Tables that are defined in the current Pipe Policy List are loaded and become active. During Job Optimizer Pipes execution you can modify the current Pipe Policy List contents and dynamically activate your changes. Dynamic activation of the Pipe Policy List means replacing all active (loaded) Pipe Policy Tables with the Pipe Policy Tables that are defined in the current Pipe Policy List. Dynamic activation of the Pipe Policy List also includes replacing the Pipe Rule Defaults with contents of the Pipe Rule defaults member (JOPDEFRL) from the control data set.

When the current Pipe Policy List has been changed you can use a Job Optimizer Pipes command or the MAINVIEW Batch Optimizer dialog to activate it. You can dynamically activate changes to the current Pipe Policy List only. To activate another Pipe Policy List, you must change the BatchPlex member PIPELIST parameter to point to the new Pipe Policy List and stop and restart Job Optimizer Pipes. From this point on, the new Pipe Policy List becomes current and can be dynamically activated.

When you activate a Pipe Policy List by issuing a command or by using the dialog, Job Optimizer Pipes responds with messages such as the following, to indicate whether the activation was successful:

```
BMC282263I RULE LOAD STARTED. TABLE=ALL
BMC282264I RULE LOAD ENDED. RC=rc
```

If the activation is not successful, Job Optimizer Pipes issues messages explaining the error and continues to use the loaded Pipe Policy Tables.

Determining Active Pipe Policy Tables and Rules

Because a Pipe Policy Table can be activated or removed from the active environment at any time, the Pipe Policy Tables that are referenced by the current Pipe Policy List might not be the ones in effect. Before activating a Pipe Policy List, you should determine which Pipe Policy Tables are active.

To determine which pipe policy tables are active, enter the F MBOP,DISPLAY RULE = ALL command from the MVS console.

For more information about the command and its output, see Chapter 11, “Job Optimizer Pipes Operator Commands.”

Activating a Pipe Policy List by Using a Command

To activate a Pipe Policy List (replace all active Pipe Policy Tables with the Pipe Policy Tables that are defined in the current Pipe Policy List) using a command, enter the F MBOP,LOAD TABLE=ALL command from the MVS console.

For more information about the command, see Chapter 11, “Job Optimizer Pipes Operator Commands.”

Activating a Pipe Policy List Using the Dialog

To activate a Pipe Policy List (replace all active Pipe Policy Tables with the Pipe Policy Tables that are defined in the current Pipe Policy List) using the dialog, complete the following steps:

Step 1 Access the MAINVIEW Batch Optimizer Objects List panel, shown in Figure 9-2 on page 9-6.

Step 2 Position the cursor in the input field to the left of the Pipe Policy List that you want to activate (member prefix PIPLST). Type **A** (Activate), and press **Enter**. The Policy Activation BatchPlex Name pop-up panel is displayed.

Step 3 Type the name of a BatchPlex on which to activate the Pipe Policy List and press **Enter**. A request is passed to Job Optimizer Pipes to activate the Pipe Policy List. The Pipe Policy List will be activated in all Job Optimizer Pipes address spaces that are active in Global mode in all systems in the sysplex.

Pipe Policy List Entry Options

This section describes the purpose of each Pipe Policy List entry option, its valid settings, and its default values. The options are:

- member name suffix for the Pipe Policy Table
- data set name that contains the pipe policy table member
- descriptive comment that is maintained on the PIPEPOLICY statement of the Pipe Policy Table

This section describes fields that identify a Pipe Policy Table within a Pipe Policy List.

Name Suffix

The value that you specify in the Name suffix field is used to establish a name for a new or existing Pipe Policy Table. The MAINVIEW Batch Optimizer dialog will attach this suffix to the characters *PIPPOL* to construct a Pipe Policy Table name. The name that you add can be a new or existing Pipe Policy Table; however, it cannot already exist in the Pipe Policy List. The following table lists the values, default, and keyword for this field.

Item	Description
Values	A two-byte value that adheres to member name rules.
Default	None
Keyword	RULEMEM=PIPPOL{two characters consisting of *A–Z*, 0-9, $, # or @}

Data Set Name

The value that you specify in the Data set name field identifies the partitioned data set containing the Pipe Policy Table member which is defined by the Name suffix field. The following table lists the values, default, and keyword for this field.

Item	Description
Values	A value that adheres to data set name rules.
Default	The currently accessed control data set.
Keyword	RULELIB={valid data set name}

BMC Software, Inc., Confidential and Proprietary Information

Comment

The value that you specify in the Comment field serves as comment text which is included on the line containing the PIPEPOLICY keyword in the Pipe Policy Table. The following table lists the values, default, and keyword for this field.

Item	Description
Values	22 characters
Default	None
Keyword	PIPEPOLICY comment text

Registering a Pipe Policy Table

A Pipe Policy Table defines a group of Pipe Policy rules. This section describes how to use the MAINVIEW Batch Optimizer dialog to perform Pipe Policy Table registration tasks.

The dialog includes online Help for each panel. It also provides input field help to describe operational considerations and valid field values. For more information about accessing the MAINVIEW Batch Optimizer online Help, see Appendix B, “Navigating the User Interface.”

You created a sample Pipe Policy Table containing a single rule during the customization process in preparation for executing the installation verification jobs. You can tailor this table or add new Pipe Policy Tables in which you include additional rules to meet the needs of your site.

The MAINVIEW Batch Optimizer dialog provides two methods for editing a Pipe Policy Table:

- Use the edit line command to select a PIPPOL*xx* member from the MAINVIEW Batch Optimizer Objects List panel, as shown in Figure 9-2 on page 9-6. The panel flow is shown in Figure 9-6 on page 9-14.

- Select a PIPLST*xx* member from the MAINVIEW Batch Optimizer Objects Lists panel, shown in Figure 9-2 on page 9-6. Use the edit line command to select a PIPPOL*xx* member from the Pipe Policy List panel, shown in Figure 9-3 on page 9-7. The panel flow is shown in Figure 9-7 on page 9-15.

Note: Editing or viewing a Pipe Policy Table is explained in this chapter. For more information on creating a new Pipe Policy Table, see Appendix B, “Navigating the User Interface.”

Figure 9-6 shows the panel flow when editing the Pipe Policy Table from the MAINVIEW Batch Optimizer Objects List.

Figure 9-6 Pipe Policy Table Registration–Dialog Panel Flow

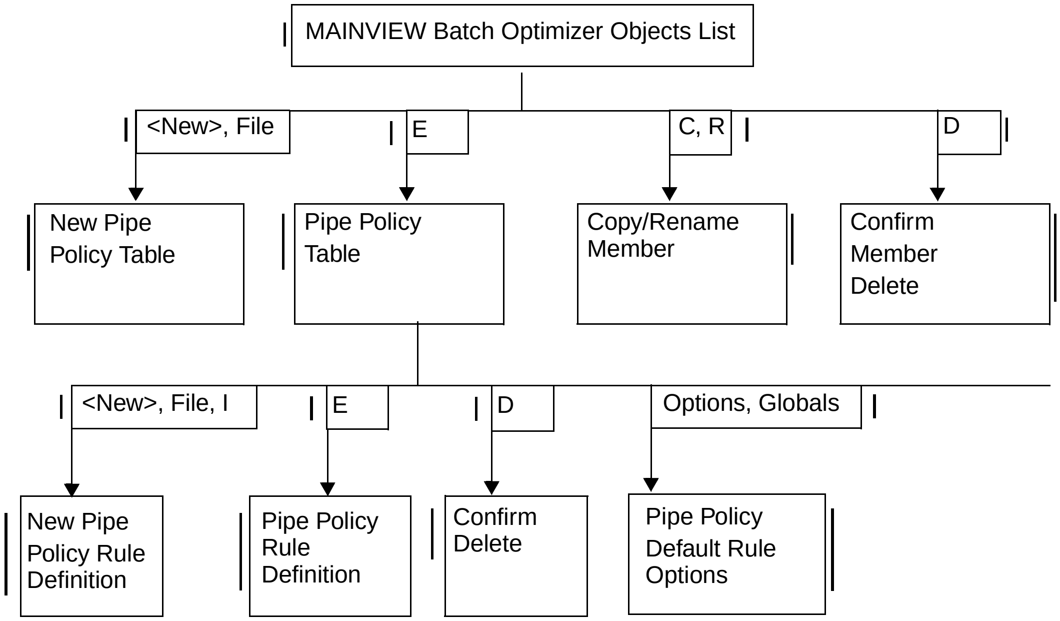
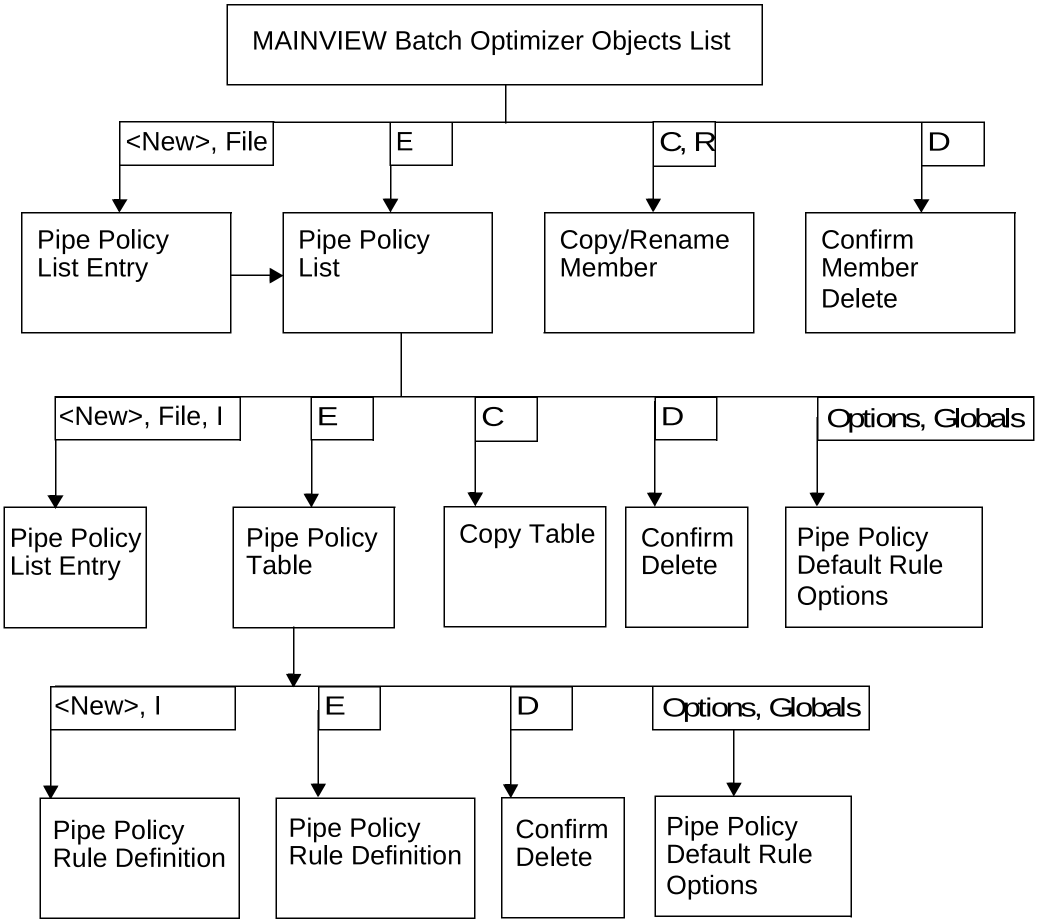


Figure 9-7 Pipe Policy Table Registration through Pipe Policy List



Editing or Viewing a Pipe Policy Table

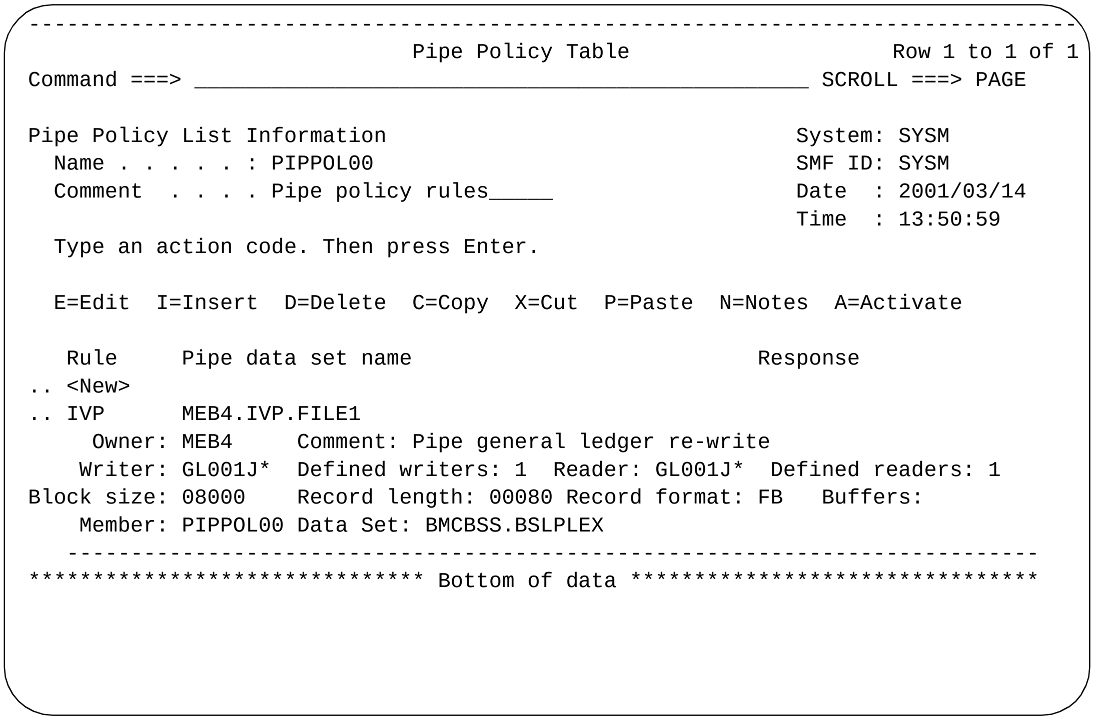
You can edit or view the information in a Pipe Policy Table. To edit or view information for a Pipe Policy Table, complete the following steps:

Step 1 Access the MAINVIEW Batch Optimizer Objects List panel, shown in Figure 9-2 on page 9-6, or the Pipe Policy List panel, shown in Figure 9-3 on page 9-7.

Step 2 Position the cursor in the input field that is left of the Pipe Policy Table that you want to edit. Type **E** (Edit), and press **Enter**.

The Pipe Policy Table panel is displayed. Figure 9-8 on page 9-16 shows an example of the Pipe Policy Table panel, listing a single rule definition.

Figure 9-8 Pipe Policy Table Panel



```
----------------------------------------------------------------------------------
                              Pipe Policy Table                    Row 1 to 1 of 1
Command ===> ________________________________________________ SCROLL ===> PAGE

Pipe Policy List Information                                System: SYSM
  Name . . . . . : PIPPOL00                                 SMF ID: SYSM
  Comment  . . . . Pipe policy rules_____                   Date  : 2001/03/14
                                                            Time  : 13:50:59
  Type an action code. Then press Enter.

  E=Edit  I=Insert  D=Delete  C=Copy  X=Cut  P=Paste  N=Notes  A=Activate

   Rule     Pipe data set name                           Response
.. <New>
.. IVP      MEB4.IVP.FILE1
     Owner: MEB4     Comment: Pipe general ledger re-write
    Writer: GL001J*  Defined writers: 1  Reader: GL001J*  Defined readers: 1
Block size: 08000    Record length: 00080 Record format: FB   Buffers:
    Member: PIPPOL00 Data Set: BMCBSS.BSLPLEX
   ----------------------------------------------------------------------------
******************************* Bottom of data ********************************
```

Step 3 To make changes to any entry field listed on this panel, position the cursor to the field, clear the field, then type in the new text.

Step 4 For more information about making changes to the Pipe Rule defaults, see “Editing or Viewing Pipe Rule Defaults” on page 9-18.

Step 5 For details on how to make changes to the pipe rules defined in the Pipe Policy Table, see “Editing or Viewing a Pipe Policy Rule” on page 9-23.

Activating a Pipe Policy Table

When you use the MAINVIEW Batch Optimizer dialog to edit a Pipe Policy Table, you must activate (load) the table to have your changes take effect. Saving an edited Pipe Policy Table does not cause the changes to take effect.

You can use a Job Optimizer Pipes command or the MAINVIEW Batch Optimizer dialog to activate (load) a Pipe Policy Table. If a previous copy of the table is already active, it is replaced by the updated table. Otherwise, the table is added to the active tables and remains active for the current execution of Job Optimizer Pipes only. When you restart the Job Optimizer Pipes address space, the Pipe Policy Tables in the current Pipe Policy List (the Pipe Policy List that is identified in the BatchPlex definition PIPELIST parameter) become active again. To make a Pipe Policy Table permanently active, you must add it to the current Pipe Policy List.

You can display the active Pipe Policy Tables and rules. For more information, see “Determining Active Pipe Policy Tables and Rules” on page 9-11.

When you activate a Pipe Policy Table by issuing a command or by using the dialog, Job Optimizer Pipes responds with the following messages to the console:

```
BMC282263I RULE LOAD STARTED. TABLE=table-name
BMC282264I RULE LOAD ENDED. RC=rc
```

Activating a Pipe Policy Table by Using a Command

To activate (load) a Pipe Policy Table by using a command, enter the F MBOP,LOAD TABLE=member,LIB=data set command from the MVS console.

The syntax of the command can be found in “Manual Loading of Pipe Rules to the Active Environment” on page 11-10.

Activating a Pipe Policy Table by Using the Dialog

To activate (load) a Pipe Policy Table by using the dialog, complete the following steps:

Step 1 Access the MAINVIEW Batch Optimizer Objects List panel, shown in Figure 9-2 on page 9-6, or the Pipe Policy List panel, shown in Figure 9-3 on page 9-7.

Step 2 Position the cursor in the input field left of the Pipe Policy Table that you want to activate (member prefix *PIPPOL*). Type **A** (Activate), and press **Enter**. The Policy Activation BatchPlex Name pop-up panel is displayed.

Step 3 Type the name of a BatchPlex on which to activate the Pipe Policy Table, and press **Enter**. A request is passed to Job Optimizer Pipes to activate the Pipe Policy Table. The Pipe Policy Table is activated in all Job Optimizer Pipes address spaces that are active in Global mode in all systems in the sysplex.

Editing or Viewing Pipe Rule Defaults

Pipe Rule Defaults are values used by Job Optimizer Pipes for pipe rule definition fields left blank. These values are saved in the Pipe Default Rule table JOPDEFRL. You can review or modify these values.

To edit or view the pipe default rule values, complete the following steps:

Step 1 Access the Pipe Policy List panel, shown in Figure 9-3 on page 9-7, the Pipe Policy Table panel, shown in Figure 9-8 on page 9-16, or the Pipe Policy Rule Definition Panel, shown in Figure 9-9 on page 9-22. For more information about accessing these panels, see

- “Editing or Viewing a Pipe Policy List” on page 9-6
- “Editing or Viewing a Pipe Policy Table” on page 9-16
- “Editing or Viewing a Pipe Policy Rule” on page 9-23

Step 2 Choose one of the following methods:

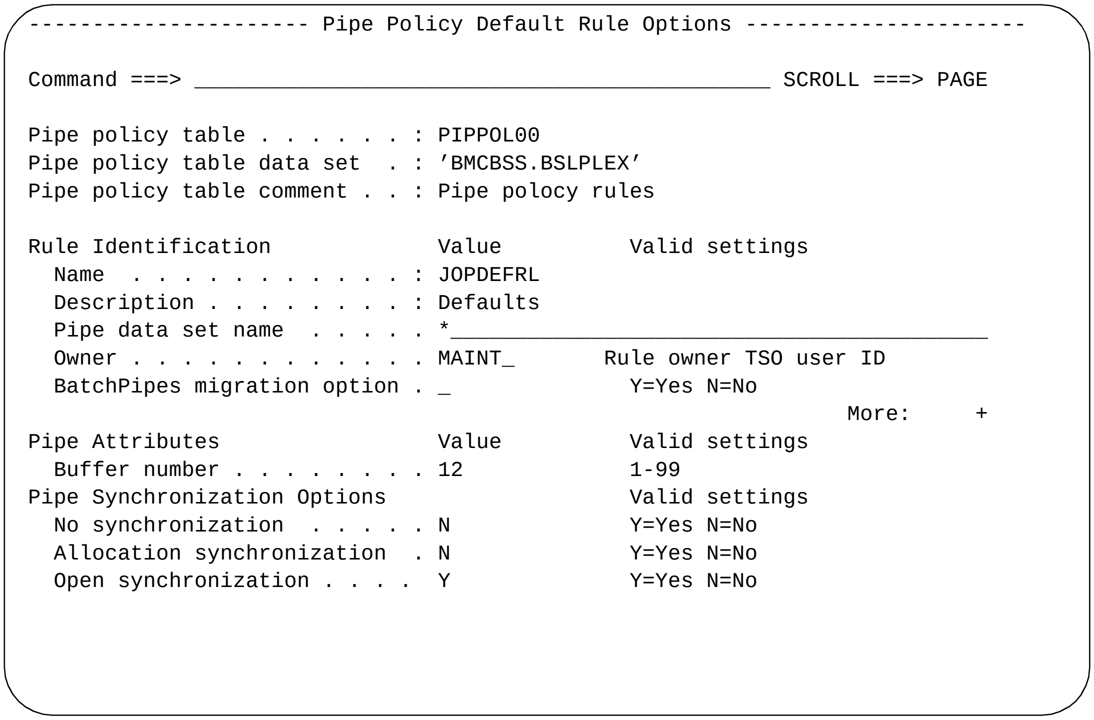
- Type the command Globals in the command line and press **Enter**.

- Position the cursor on the action bar Options item, and press **Enter**. The Options pull-down is displayed. Type **6** (Pipe policy rule defaults), and press **Enter**.

The Pipe Policy Default Rule Options pop-up panel, shown in Figure 9-9, is displayed. You can specify or change values that serve as the default field values for rule definition items left blank. The information you specify or change is the same as that found on the Pipe Policy Rule Definition panel, shown in Figure 9-10 on page 9-24.

The customization process established a set of default rule values to support the installation verification process.

Figure 9-9 Pipe Policy Default Rule Options Pop-up Panel



```
--------------------- Pipe Policy Default Rule Options ---------------------

Command ===> _____________________________________________ SCROLL ===> PAGE

Pipe policy table . . . . . . : PIPPOL00
Pipe policy table data set  . : ’BMCBSS.BSLPLEX’
Pipe policy table comment . . : Pipe polocy rules

Rule Identification             Value          Valid settings
  Name  . . . . . . . . . . . : JOPDEFRL
  Description . . . . . . . . : Defaults
  Pipe data set name  . . . . . *__________________________________________
  Owner . . . . . . . . . . . . MAINT_       Rule owner TSO user ID
  BatchPipes migration option . _              Y=Yes N=No
                                                                More:     +
Pipe Attributes                 Value          Valid settings
  Buffer number . . . . . . . . 12             1-99
Pipe Synchronization Options                   Valid settings
  No synchronization  . . . . . N              Y=Yes N=No
  Allocation synchronization  . N              Y=Yes N=No
  Open synchronization . . . .  Y              Y=Yes N=No
```

Step 3 To change any input field, position the cursor in the field, clear the current value if any, and type a new value. Repeat this step for each field you want to modify.

Step 4 Press **Enter** to save the changes, or press **F12** (Cancel) to discard your changes. One of the following panels is displayed:

- the Pipe Policy List panel, shown in Figure 9-3 on page 9-7
- the Pipe Policy Table panel, shown in Figure 9-8 on page 9-16
- the Pipe Policy Rule Definition Panel, shown in Figure 9-10 on page 9-24

You can commit changes to the control data set.

Step 5 Commit the changes to the control data set. The changed default rule data is saved in member JOPDEFRL of the control data set when the Pipe Policy Table or Pipe Policy List is saved.

Activating the Pipe Rule Defaults

Changing the Pipe Rule defaults does not cause the changes to take effect. You can use one of the following methods to activate the changes:

- activate the current Pipe Policy List

  For more information, see “Activating a Pipe Policy List” on page 9-10.

- activate the Pipe Policy Table JOPDEFRL

  For more information about activating a Pipe Policy Table, see “Activating a Pipe Policy Table” on page 9-17.

Modifying Pipe Policy Table Entries

This section describes administration options to manipulate the Pipe Policy Table entries.

Adding a New Rule Definition

To add a new rule definition to the top of the list, choose one of the following methods:

- Type **New** on the Command Line, and press **Enter**.

- If the top of the panel contains an Action bar, select **1** (New) from the File pull-down and press **Ente**r.

- Position the cursor to the left of the <new> rule definition, type E (Edit), and press **Enter.**

The Pipe Policy Rule Definition panel is displayed. Edit the new definition.

Inserting a New Rule Definition

To insert (similar to add, but in your choice of list position) a new rule definition, complete the following steps:

Step 1 Decide after which rule definition you want the new one to be placed.

Step 2 Position to the left of that rule definition.

Step 3 Type **I** (Insert).

Step 4 Press **Enter**.

The Pipe Policy Rule Definition panel is displayed. Edit the new definition.

Copying and Pasting a Rule Definition

To copy (and paste) a rule definition, complete the following steps:

Step 1 Position the cursor to the left of the rule definition you wish to copy.

Step 2 Type **C** (Copy).

Step 3 Press **Enter**. The message, “Definition copied to clipboard,” is displayed.

Step 4 Type **P** (Paste) to the left of the line after which you want to place the copied rule.

Step 5 Press **Enter**. The definition is moved from the clipboard to the selected position on the list.

Moving a Rule Definition

To move (cut and paste) an entry, complete the following steps:

Step 1 Position the cursor to the left of the rule definition that you want to move.

Step 2 Type **X** (Cut).

Step 3 Press **Enter**. The message “Definition cut to clipboard and deleted from definition list” is displayed.

Step 4 Type **P** (Paste) to the left of the line following which you want to place the cut entry.

Step 5 Press **Enter**. The rule definition is moved from the clipboard to the selected position on the list.

Deleting a Rule Definition

To delete an entry, complete the following steps:

Step 1 Position the cursor to the left of the rule definition that you want to delete.

Step 2 Type **D** (Delete).

Step 3 Press **Enter**. The Confirm Delete pop-up panel is displayed. This step lets you confirm the delete request before removing the rule definition from the Pipe Policy Table.

Saving Your Changes

You can save your changes by using one of the following methods:

- Type **Save** on the Command Line, and press **Enter.** When the change is saved, the Pipe Policy Table panel is displayed again.

- If the top of the panel contains an Action bar, select the File pull-down choice 2 (Save Pipe Policy Table to disk). When the change is saved, the Pipe Policy Table panel is displayed.

- Press **F12** (Cancel). The Confirm Cancel pop-up panel is displayed. Type **1** (Save changes to disk and cancel), and press **Enter**. The dialog saves the changed pipe policy members and re-displays the Pipe Policy List panel or the MAINVIEW Batch Optimizer Objects List panel.

- Press **F3** (Exit). The Confirm Exit pop-up panel is displayed. Type **1** (Save changes to disk and exit), and press **Enter**. The dialog saves the changed pipe policy members and displays the MAINVIEW Batch Optimizer Objects List panel.

BMC Software, Inc., Confidential and Proprietary Information

Editing or Viewing a Pipe Policy Rule

You can review or modify the following information saved for a pipe policy rule definition:

- Rule Identification options
- Pipe Attributes options
- Pipe Participant Synchronization options
- Pipe Participant Maximum Wait Time options
- Pipe Writer Job options
- Pipe Reader Job options

For more information about the available options, see "Pipe Rule Options" on page 9-27.

To edit or view a pipe policy rule definition, complete the following steps:

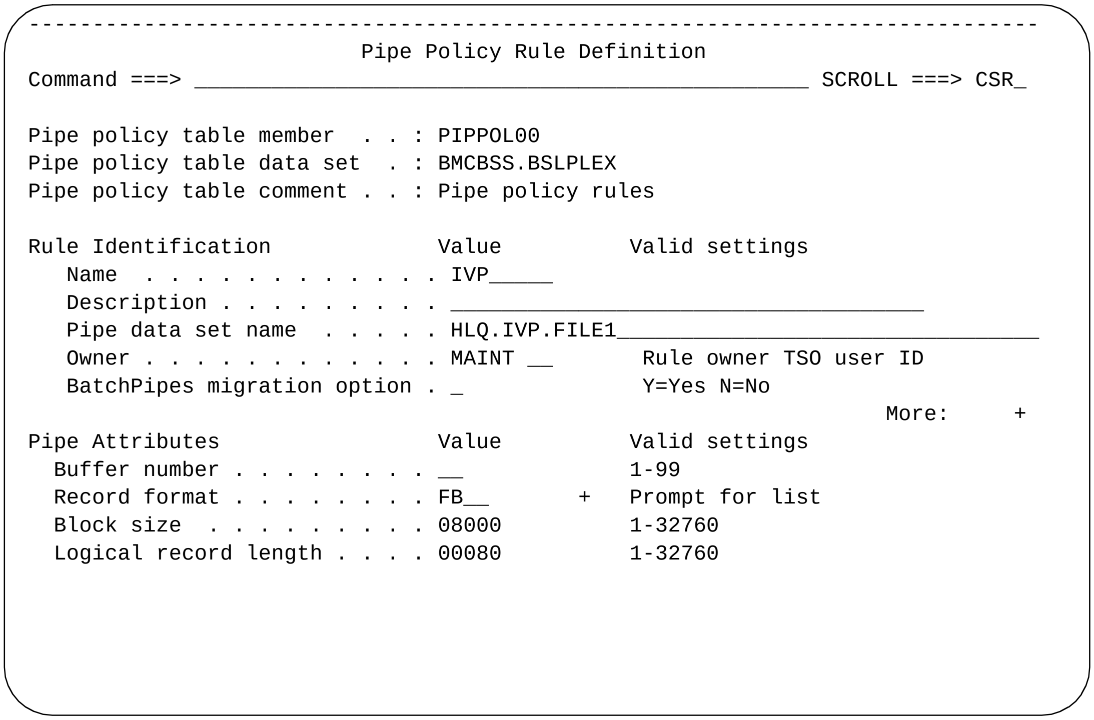
Step 1 Access the Pipe Policy Table panel. For more information about accessing this panel, see "Editing or Viewing a Pipe Policy Table" on page 9-16 panel.

Step 2 Position the cursor in the input field left of the pipe rule you wish to edit. Type **E** (Edit), and press **Enter**. The dialog displays the Pipe Policy Rule Definition panel, where you can specify or change such rule information as:

- Rule identification information such as the rule name, an optional description, the pipe data set name, and the rule owner ID
- Pipe attributes
- Pipe participant synchronization options
- Pipe participant maximum wait time options
- Pipe writer(s) and reader(s) information

Figure 9-10 on page 9-24 shows an example of the Pipe Policy Rule Definition panel, which was created during the customization process. HLQ would be replaced with the high level qualifier specified during customization. Options left blank use values specified in the Pipe Rule defaults. Press **F6** (Defaults) to display the default value for a particular field. For information on other available dialog options, see Appendix B, "Navigating the User Interface." For more information on modifying the Pipe Rule defaults, see "Editing or Viewing Pipe Rule Defaults" on page 9-18.

Figure 9-10 Pipe Policy Rule Definition Panel



```
-------------------------------------------------------------------------------
                          Pipe Policy Rule Definition
Command ===> ________________________________________________ SCROLL ===> CSR_

Pipe policy table member  . . : PIPPOL00
Pipe policy table data set  . : BMCBSS.BSLPLEX
Pipe policy table comment . . : Pipe policy rules

Rule Identification             Value          Valid settings
   Name  . . . . . . . . . . . . . IVP_____
   Description . . . . . . . . . _____________________________________
   Pipe data set name  . . . . . HLQ.IVP.FILE1_________________________________
   Owner . . . . . . . . . . . . . MAINT __       Rule owner TSO user ID
   BatchPipes migration option . _              Y=Yes N=No
                                                                   More:     +
Pipe Attributes                 Value          Valid settings
  Buffer number . . . . . . . . __             1-99
  Record format . . . . . . . . FB__       +   Prompt for list
  Block size  . . . . . . . . . 08000          1-32760
  Logical record length . . . . 00080          1-32760
```

Step 3 Make changes to any entry field by positioning the cursor to the field and typing the new value. Blank out any characters that remain from the previously displayed value. Leave blank any field for which you want to use the default rule value. Repeat this step for each field you want to edit.

Step 4 For the Record format field, you can prompt for a list of acceptable values by positioning the cursor to the field and pressing **F4** (Prompt). For more information on how to access the Prompt facility, see Appendix B, “Navigating the User Interface.”

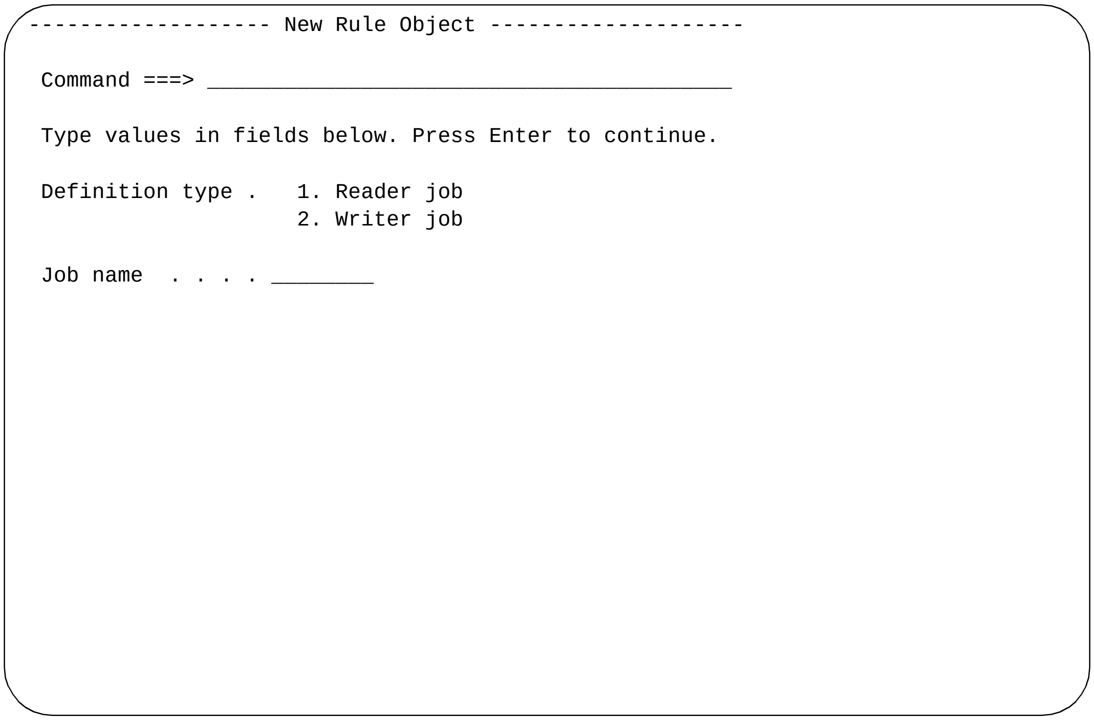
Step 5 To update the Writer jobs and Reader jobs fields with more names, perform the following steps:

5.A Choose one of the following methods:

- Select File pull-down choice 1 (New writer/reader job), or type New on the Command Line to display the New Rule Object panel, as shown in Figure 9-11 on page 9-25. Continue with step 5.B.

- Position the cursor to either the Reader Job field or Writer Job field, press **F4** (Prompt), and select <New> from the displayed list. This option will display Rule Reader Job Specifications panel, as shown in Figure 9-12 on page 9-26, or the Rule Writer Job Specifications panel, as shown in Figure 9-13 on page 9-26. Continue with step 5.D.

Figure 9-11 New Rule Object Panel

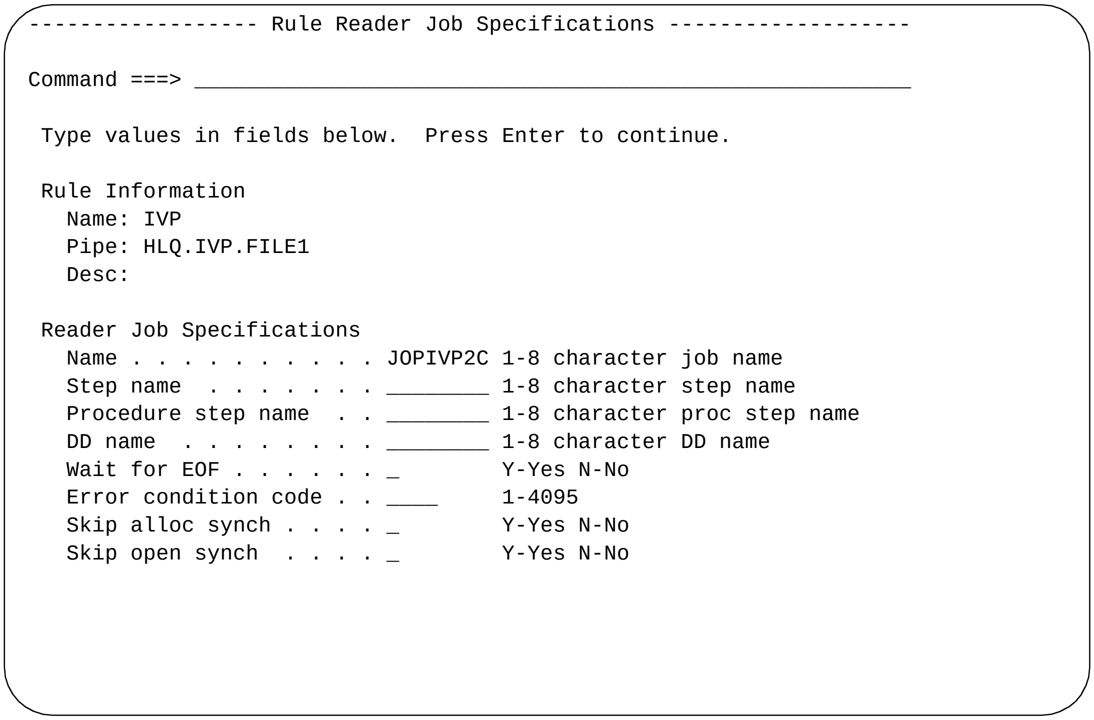


```
------------------ New Rule Object -------------------

 Command ===> _________________________________________

 Type values in fields below. Press Enter to continue.

 Definition type .   1. Reader job
                     2. Writer job

 Job name  . . . . ________
```

5.B From the New Rule Object panel, select **1** (Reader job) or **2** (Writer job) and type a name in the Job Name field.

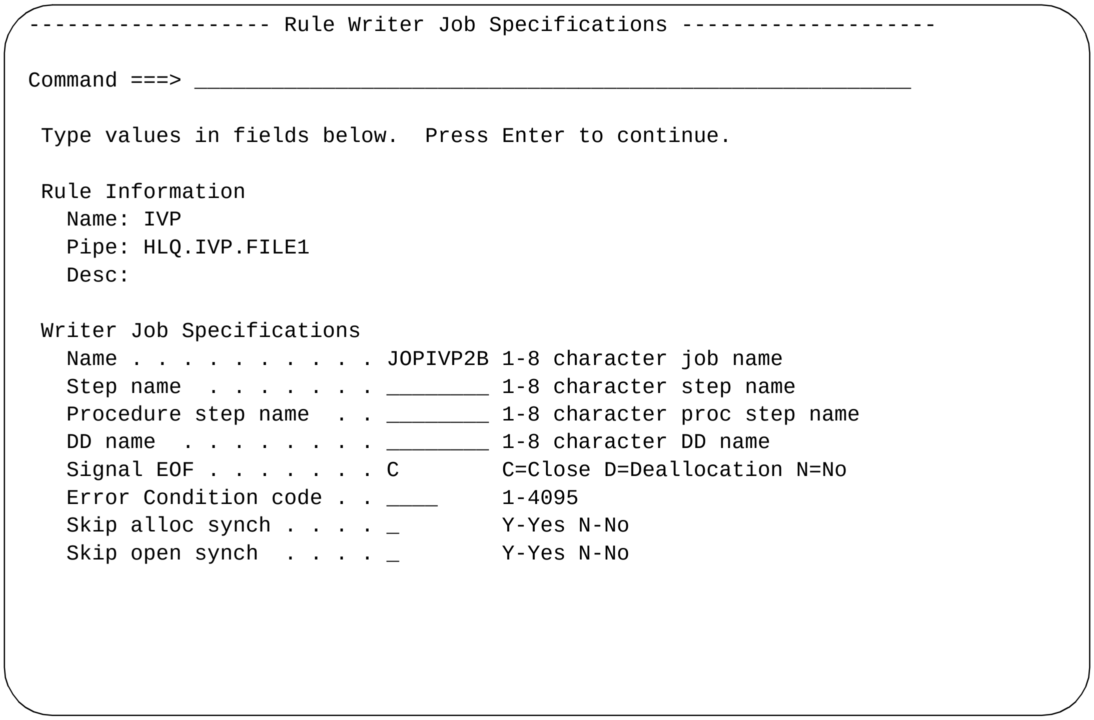
5.C Press **Enter** to display either the Rule Reader Job Specifications panel, shown in Figure 9-12 on page 9-26 or the Rule Writer Job Specifications panel, shown in Figure 9-13 on page 9-26.

Figure 9-12 Rule Reader Job Specifications Panel



```
------------------ Rule Reader Job Specifications -------------------

Command ===> ________________________________________________________

 Type values in fields below.  Press Enter to continue.

 Rule Information
   Name: IVP
   Pipe: HLQ.IVP.FILE1
   Desc:

 Reader Job Specifications
   Name . . . . . . . . . . JOPIVP2C 1-8 character job name
   Step name  . . . . . . . ________ 1-8 character step name
   Procedure step name  . . ________ 1-8 character proc step name
   DD name  . . . . . . . . ________ 1-8 character DD name
   Wait for EOF . . . . . . _        Y-Yes N-No
   Error condition code . . ____     1-4095
   Skip alloc synch . . . . _        Y-Yes N-No
   Skip open synch  . . . . _        Y-Yes N-No
```

Figure 9-13 Rule Writer Job Specifications Panel



```
------------------- Rule Writer Job Specifications --------------------

Command ===> ________________________________________________________

 Type values in fields below.  Press Enter to continue.

 Rule Information
   Name: IVP
   Pipe: HLQ.IVP.FILE1
   Desc:

 Writer Job Specifications
   Name . . . . . . . . . . JOPIVP2B 1-8 character job name
   Step name  . . . . . . . ________ 1-8 character step name
   Procedure step name  . . ________ 1-8 character proc step name
   DD name  . . . . . . . . ________ 1-8 character DD name
   Signal EOF . . . . . . . C        C=Close D=Deallocation N=No
   Error Condition code . . ____     1-4095
   Skip alloc synch . . . . _        Y-Yes N-No
   Skip open synch  . . . . _        Y-Yes N-No
```

5.D Enter the information on either of these panels to define the pipe participant job name and other options. Fields you leave blank will use the default rule value.

5.E Press **Enter** to save the pipe participant job changes, or press **F12** (Cancel) to discard your changes.

Step 6 Press **Enter** to save the changes, or press **F12** (Cancel) to discard your changes. The dialog displays the Pipe Policy Table panel, shown in Figure 9-8 on page 9-16, where you can then commit changes to the control data set.

Step 7 Commit the changes to the control data set. For more information on saving changes to a Pipe Policy Table, see "Saving Your Changes" on page 9-22.

Pipe Rule Options

Job Optimizer Pipes uses rules to allow you to specify data set candidates for job-to-job piping or for BatchPipes Phased Migration processing and to specify pipe processing options. You provide these options via one of two places:

- Pipe Policy Rule Definition panel, which you use to define a pipe rule

- Pipe Policy Default Rule Options panel, which you use to define default rule values for each option

This section describes the purpose of each pipe rule option. The dialog groups these options in the following panel sections:

- Rule Identification options
- Pipe Attributes options
- Synchronization options
- Maximum Wait Time options
- Writer options
- Reader options

Character Masking

Several fields support character masking. If a field supports masking, the support is noted. Table 9-1 lists mask character functions.

Table 9-1 Character Masking

Character	Function
*	represents any number of characters, including no characters
?	represents any one character

Example

Assume the following names exist: A3, M, M01, M03, M13, M23, M33, M103, M2243.

Table 9-2 lists the results of using different character masking specifications.

Table 9-2 Character masking examples

Entry	Matching Values
*	All the above values
M?3	M03, M13, M23, M33
M??3	M103
M*3	M03, M13, M23, M33, M103, M2243
M*	All preceding values except A3. Because the last character is *, *M* is treated as a prefix.

Rule Identification

Rule Identification options let you specify the following information:

- user-specified name for the rule
- free text description of the rule
- name of the data set that is to be replaced by a pipe or name of BatchPipes pipe to be migrated to Job Optimizer Pipes
- ID of the rule owner
- BatchPipes migration option information

BMC Software, Inc., Confidential and Proprietary Information

Name

In this field, specify a name for the rule. The rule name does not have to be unique within a Pipe Policy Table. However, BMC Software recommends that unique names be used since this simplifies rule management activities such as Holding and Releasing individual rules. The following table lists the values and default for this field.

Item	Description
Values	A name that is 1-8 characters in length. The name must adhere to member name standards.
Default	None

Description

In this field, optionally specify a free form text description of the rule. You can use this text to help manage your pipe rules by further identifying them. The following table lists the values and default for this field.

Item	Description
Values	Up to 46 characters of text.
Default	None

Pipe Data Set Name

In this field, you provide the name of the data set to be replaced by a pipe or the name of the BatchPipe pipe to be migrated to Job Optimizer Pipes.

- When replacing a Generation Data Set (GDG) with a pipe, specify the GDG Base name followed by .G*, to ensure that only the GDG members are replaced by the pipe. For example, if the GDG Base name is PROD.REPORT, specify the data set name in the rule as PROD.REPORT.G*. Job Optimizer Pipes uses a generation number of G0000V00 instead of the real generation number for the pipe name. If you also elect to create a physical file, Job Optimizer Pipes creates the file using the real generation number.
- When specifying the name of the BatchPipes pipe to be migrated to Job Optimizer Pipes, see “BatchPipes Migration Option” on page 9-30.

The following table lists the values and defaults for this field.

Item	Description
Values	A data set name that is 1-44 characters in length. It must adhere to JCL data set name standards. Mask characters can be used.
Default	None

Owner

This field contains the User ID identified with the creation of a pipe rule. This user ID is considered the owner of the rule. The following table lists the values and default for this field.

Item	Definition
Values	A name that is 1-8 characters in length. It must adhere to TSO User ID standards.
Default	TSO User ID currently executing the dialog

BatchPipes Migration Option

In this field, you specify whether the rule will be used for BatchPipes Phased Migration or for job-to-job pipes. A rule defined for BatchPipes Phased Migration is called a Pipe Policy Migration Rule. When a Pipe Policy Migration Rule is defined, only the rule identification section and the selection criteria in the writer and reader sections should be filled. All other fields are ignored. For more information about BatchPipes migration, see Appendix E, “Migrating from BatchPipes to Job Optimizer Pipes.” The following table lists the values and default for this field.

Item	Description
Values	This option has the following possible values: • Y–This is a Pipe Policy Migration Rule. • N–This is a Pipe Policy Rule defining a data set that is a candidate for job-to-job pipe processing.
Default	N

Pipe Attributes

The Pipe Attributes section allows specifying various attributes of the pipe data set. These values include:

- number of buffers
- record format

- block size
- logical record length

Buffer Number

In this field, specify the number of buffers to allocate for a pipe. Each buffer can contain one data block. This parameter determines the number of data blocks that can be in the pipe at any one time; it does not limit the total quantity of data that passes through the pipe. BMC Software recommends that you specify a value from 5 to 15. If there is more than one reader, and the value in the Reader data field is set to N (for Next, so that each reader gets the next available unread block), the Buffer number value should be set somewhat higher. The following table lists the values and default for this field.

Item	Description
Values	A numeric value between 1 and 99.
Default	The value specified in JOPDEFRL.

Record Format

In this field, you can specify the pipe record format. The following sources are searched (in the order listed below) to find the record format value of the pipe:

- RECFM that is specified for the pipe or the replaced data set within the application (for example, if DCB is specified)

- RECFM that is specified in the JCL for the pipe or the replaced data set

- Value that is specified in this parameter in the rule

The record format must be compatible with the pipe record length and block size. The pipe record format must be identical for all the participants in the pipe. If a file is created with the pipe, the file will get the same DCB attributes as the pipe.

Note: BMC Software recommends leaving this field empty so the pipe will have the same attributes as were used for the replaced data set.

The following table lists the values and default for this field.

Item	Description
Values	This option has the following possible values: • F–Fixed • FB–Fixed blocked • FA–Fixed ISO/ANSI control • FM–Fixed machine control • FS–Fixed standard • FBA–Fixed blocked ISO/ANSI control • FBM–Fixed blocked machine control • FBS–Fixed blocked standard • FSA–Fixed standard ISO/ANSI control • FSM–Fixed standard machine control • FBSA–Fixed blocked standard ISO/ANSI control • FBSM–Fixed blocked standard machine control • V–Variable • VB–Variable blocked • VA–Variable ISO/ANSI control • VM–Variable machine control • VBA–Variable blocked ISO/ANSI control • VBM–Variable blocked machine control
Default	There is no default for the writer. The default Record Format for the reader is copied from the pipe after the pipe was initialized with the writer’s values.

Block Size

In this field, you can specify the pipe block size. The following sources are searched (in the order listed below) to find the block size value of the pipe:

- Block size that is specified for the pipe or the replaced data set within the application (for example, if DCB is specified)
- Block size that is specified in the JCL for the pipe or the replaced data set
- Value that is specified in this parameter in the rule

The block size must be compatible with the pipe record length and record format. The pipe block size must be identical for all the participants in the pipe. If a file is created with the pipe, the file will get the same DCB attributes as the pipe.

Note: BMC Software recommends leaving this field empty so the pipe will have the same attributes as were used for the replaced data set.

The following table lists the values and default for this field.

Item	Description
Values	A numeric value between 0 and 32760.
Default	For a writer, the default block size is calculated when Logical Record Length (LRECL) and Record Format (RECFM) are specified in one of the sources listed above. Otherwise, no default block size is calculated. The default block size calculation depends on the Create File Writer option specification. If a file is not created, the default block size is calculated as follows: • For fixed blocked records, the largest block size less than or equal to 32760 that is an integer multiple of the LRECL. • For variable blocked records, the value 32760. • For fixed unblocked records, the LRECL value. • For variable unblocked records, the LRECL value plus 4. If a file is created, the default block size is calculated as follows: • For blocked records, fixed or variable, the System Determined Blocksize (SDB) that is calculated by the system for the file. • For fixed unblocked records, the LRECL value. • For variable unblocked records, the LRECL value plus 4. The default block size for a reader is copied from the pipe after the pipe is initialized with the writers' values.

Logical Record Length

In this field, you can specify the pipe logical record length. The following sources are searched (in the order listed below) to find the logical record length value of the pipe:

- LRECL specified for the pipe or the replaced data set within the application (for example, if DCB is specified)

- LRECL specified in the JCL for the pipe or the replaced data set

- Value specified in this parameter in the rule

The record length must be compatible with the pipe record format and block size. The pipe record length must be identical for all the participants in the pipe. If a file is created with the pipe, the file will get the same dcb attributes as the pipe.

Note: BMC Software recommends leaving this field empty so the pipe will have the same attributes as were used for the replaced data set.

The following table lists the values and default for this field.

Item	Description
Values	A numeric value between 1 and 32760.
Default	There is no default for the writer. The default Logical Record Length for the reader is copied from the pipe after the pipe was initialized with the writer's values.

Synchronization Options

Synchronization options identify points during processing at which Job Optimizer Pipes holds all participants for synchronization before processing continues. The synchronization points are data set allocation, data set open, data set close, and data set de-allocation.

The synchronization definition fields are treated as one option. The default for this option is taken from JOPDEFRL. The default values from JOPDEFRL are used when all synchronization fields are left empty. Setting a value in one of these fields in the rule is considered as setting a value for the synchronization option. When this setting is made, the values from JOPDEFRL are not used and empty fields are treated as N. Therefore, when there is a need to change the setting of one of these fields, specify values for all synchronization points are required.

Example

Y is specified in synchronization at open and synchronization at close in JOPDEFRL. Specifying Y in synchronization at allocation in the rule will result in synchronization at allocation only. JOPDEFRL will not be used for synchronization option setting (because a value for this option is specified in the rule) and the empty fields will be treated as N.

If synchronization at allocation is required in addition to the specifications in JOPDEFRL (in this example, open and close), values should be set in all three fields in the rule.

No Synchronization

In this field, specify whether no synchronization should occur. The following table lists the values and default from this field.

Item	Description
Values	This option has the following possible values: • Y–Job Optimizer Pipes will not perform synchronization. With this option, you cannot specify the value Y for any other synchronization option field. • N–Job Optimizer Pipes will perform synchronization based upon the other synchronization option fields. **Note**: A Value of N must be set for this field if Reader Data value is set to A (All) or if Writer Error Condition Code and/or Reader Error Condition Code values are set with a value other than 0.
Default	See “Synchronization Options” on page 9-34.

Allocation Synchronization

In this field, you can specify whether Job Optimizer Pipes holds processing of each participant allocating the pipe until all participants have allocated the pipe. The following table lists the values and default for this field.

Item	Description
Values	This option has the following possible values: • Y–Job Optimizer Pipes holds processing of a pipe participant allocating the pipe until all participants have allocated the pipe. • N–Job Optimizer Pipes does not hold processing of a pipe participant allocating the pipe.
Default	See “Synchronization Options” on page 9-34.

Open Synchronization

In this field, you can specify whether Job Optimizer Pipes holds processing of each participant opening the pipe until all participants have opened the pipe. The following table lists the values and default for this field.

Item	Description
Values	This option has the following possible values: • Y–Job Optimizer Pipes holds processing of each participant opening the pipe until all participants have opened the pipe. • N–Job Optimizer Pipes does not hold processing of a pipe participant opening the pipe. **Note**: A value of Y must be set for this field if the Reader Data Value is set with a value of A (All).
Default	See “Synchronization Options” on page 9-34.

Close Synchronization

In this field, you can specify whether Job Optimizer Pipes holds processing of each participant that closes the pipe until all pipe participants have closed the pipe. The following table lists the values and default for this field.

Item	Description
Values	This option has the following possible values: • Y–Job Optimizer Pipes holds processing of a pipe participant closing the pipe until all pipe participants have closed the pipe. • N–Job Optimizer Pipes does not hold processing of a pipe participant closing the pipe. **Note**: A value of N must be set for this field if the Writer Signals EOF value is set to D (De-allocation) for one or more writers.
Default	See “Synchronization Options” on page 9-34

Deallocation Synchronization

In this field, you can specify whether Job Optimizer Pipes holds processing of each participant that de-allocates the pipe (usually during step termination) until all pipe participants have de-allocated the pipe. The following table lists the values and default for this field.

Item	Description
Values	This option has the following possible values: • Y–Job Optimizer Pipes holds processing of a pipe participant de-allocating the pipe until all pipe participants have de-allocated the pipe. • N–Job Optimizer Pipes does not hold the processing of a pipe participant de-allocating the pipe. **Note**: A value of Y must be set for this field if the Writer Error Condition Code and/or Reader Error Condition Code is set to a value other than 0.
Default	See “Synchronization Options” on page 9-34

Maximum Wait Time Options

The Maximum Wait Time Options identify various wait times that Job Optimizer Pipes will monitor before intervening by either abending the participant or issuing a request to the operator console. You can specify values for the following wait times:

- No-operation time and the action to take if exceeded
- Input-Output wait time and the action to take if exceeded
- Synchronization wait time and the action to take if exceeded

No-Operation Wait Time Option

In this field, you can specify the maximum amount of time (in minutes) during which the pipe participant has to access the pipe. The following table lists the values and default for this field.

Item	Description
Values	A numeric value between 1 and 9999
Default	The value specified in JOPDEFRL.

No-Operation Wait Action Option

In this field, you can specify what action Job Optimizer Pipes should take when the no-operation time exceeds the value specified in the No-Operation Wait Time field. The following table lists the values and default for this field.

Item	Description
Values	This option has the following possible values: • A–Job Optimizer Pipes terminates the pipe participant abnormally. • O–Job Optimizer Pipes issues a message to the operator console requiring a reply to indicate whether Job Optimizer Pipes should abnormally terminate the pipe participant or allow it to wait for an additional amount of time.
Default	The value specified in JOPDEFRL.

Input-Output Wait Time Option

In this field, you can specify the maximum amount of time (in minutes) a participant can wait for data transfer (I/O activity). The following table lists the values and default for this field.

Item	Description
Values	A numeric value between 1 and 9999
Default	The value specified in JOPDEFRL.

Input-Output Wait Action Option

In this field, you can specify what action Job Optimizer Pipes should take when the Input-Output wait time exceeds the value specified in the Input-Output Wait Time field. The following table lists the values and default for this field.

Item	Description
Values	This option has the following possible values: • A–Job Optimizer Pipes terminates the pipe participant abnormally. • O–Job Optimizer Pipes issues a message to the operator console requiring a reply to indicate whether Job Optimizer Pipes should abnormally terminate the pipe participant or allow it to wait for an additional amount of time.
Default	The value specified in JOPDEFRL.

Synchronization Wait Time Option

In this field, you can specify the maximum amount of time (in minutes) a participant can wait for each synchronization point. For more information on synchronization points, see “Synchronization Options” on page 9-34. The following table list the value and default for this field.

Item	Description
Values	A numeric value between 1 and 9999
Default	The value specified in JOPDEFRL.

Synchronization Wait Action Option

In this field, you can specify what action Job Optimizer Pipes should take when the time spent at a Synchronization point exceeds the value specified in the Synchronization Wait Time field. The following table lists the values and default for this field.

Item	Description
Values	This option has the following possible values: • A–Job Optimizer Pipes terminates the pipe participant abnormally. • O–Job Optimizer Pipes issues a message to the operator console requiring a reply to indicate whether Job Optimizer Pipes should abnormally terminate the pipe participant or allow it to wait for an additional amount of time.
Default	The value specified in JOPDEFRL.

Writer Options

The Writer options allow you to specify the following values:

- Minimum number of writers necessary for synchronization
- Number of additional writers that can join the pipe
- Writer should create a physical data set and an optional DD name
- Writer error option
- Writer selection criteria:
  - job name
  - step name
  - procedure step name
  - DD name
- Writer participant options:
  - acceptable job step condition codes
  - End-Of-File indication processing
  - participant specific allocation/open synchronization processing

Minimum Writers Option

In this field, you can specify the minimum number of writers that must reach each synchronization point before synchronization is satisfied. The following table lists the values and default for this field.

Item	Description
Values	A numeric value between 1 and 99. **Note**: A value of 1 must be set for this field if Create File is set with a value of W.
Default	1

Additional Writers

In this field, you can specify the number of additional writers that can join the pipe. The sum of the values specified in the Minimum writers field and this field is the maximum number of writers that can write data to the pipe. The following table lists the values and default for this field.

Item	Description
Values	This option has the following possible values: • Y–Job Optimizer Pipes will allow any number of additional writers to join the pipe. • N–Job Optimizer Pipes will not allow additional writers to join the pipe. • A numeric value between 1 and 99 – Job Optimizer Pipes will allow the specified number of additional writers join the pipe. **Note**: A value of N must be set for this field if Create File is set with a value of W.
Default	N

Create File Option

In this field, you can specify whether the writer should create a physical file and write the data to the file in addition to transferring data to the pipe. The following table lists the values and default for this field.

Item	Description
Values	This option has the following possible values: • W–The writer should create a physical file. • N–The writer should not create a physical file. **Note**: A value of W can be set for this field if the Minimum Writers option is set with a value of 1 and the Additional Writers option is set to a value of N.
Default	N

Create File DD Name Option

In this field, you can specify a DD name the writer job should use for creating a physical file if you specified a value of W in the Create File field. This parameter is optional. If this option is specified, the DD statement should exist in the writer's job's JCL. If this option is not specified, Job Optimizer Pipes dynamically creates the physical file without changing the JCL. The file is created according to the parameters specified in the DD statement replaced by pipe in the writer job's JCL. The following table lists the values and default for this field.

Item	Description
Values	A DD name that is 1-8 characters in length. It must adhere to JCL standards for data definition (DD) names and it must be defined in the job step JCL.
Default	None

Writer Error Option

In this field, you can specify how Job Optimizer Pipes handles writer jobs when it encounters an error in the pipe process or failure in a pipe participant. The following table lists the values and default for this field.

Item	Description
Values	This option has the following possible values: • I–Immediate. Job Optimizer Pipes abends all writer jobs immediately when an error occurs. • G–Ignore. When an error occurs, the writer continues writing to the physical file and, optionally, to the pipe. When the process finishes, the writer ends normally. **Note**: A value of G can be set for this field if all of the following conditions exist: • There is only one writer (the Minimum Writers field is set with a value of 1 and the Additional Writers field is set with a value of N). • The Create file field value is W.
Default	The value specified in JOPDEFRL.

Writer Jobs

In this field, you can specify the name of the job that writes data to the pipe data set. This field is used as selection criteria. The following table lists the values and default for this field.

Item	Description
Values	A 1 to 8 character value that adheres to JCL job name standards. Mask characters can be used.
Default	*

Writer Step Name

In this field you can specify the name of the step that writes data to the pipe data set. This parameter is optional. When specified, this field is used as selection criteria. The following table lists the values and default for this field.

Item	Description
Values	A 1- to 8-character value that adheres to the JCL standards for step names. Mask characters can be used.
Default	None

Writer Procedure Step Name

In this field you can specify the name of the step that invokes the procedure that writes data to the pipe data set. This parameter is optional. When specified, this field is used as selection criteria. The following table lists the values and default for this field.

Item	Description
Values	A 1- to 8-character value that adheres to the JCL standards for step names. Mask characters can be used.
Default	None

Writer DD Name

In this field you can specify the name of the data definition (DD) statement that is used to access the pipe data set. This parameter is optional. When specified, this field is used as selection criteria. The following table lists the values and default for this field.

Item	Description
Values	A 1- to 8-character value that adheres to the JCL standards for DD names. Mask characters can be used.
Default	None

Writer Signals EOF

In this field, you can specify whether a writer job signals an End-Of-File (EOF) condition and, if yes, when should it signal the condition. the following table lists the values and default for this field.

Item	Description
Values	This option has the following possible values: • C–The writer job should send an End-Of-File (EOF) signal to the pipe when it closes the pipe. • D–The writer job should send an End-Of-File (EOF) signal to the pipe when it de-allocates the pipe. • N–The writer job should not send an End-Of-File (EOF) signal to the pipe. **Note**: A value of D cannot be used for this field if the Close Synchronization option is set to Y.
Default	The value specified in JOPDEFRL.

Writer Error Condition Code

In this field, you can specify the lowest step condition code that Job Optimizer Pipes will treat as an error. This parameter is optional. When specified, Job Optimizer Pipes will use the step completion code to determine if there is an error in the pipe process. If the step writing to the pipe ends with a completion code equal to or higher than the value set for this field, Job Optimizer Pipes treats this as an error in the pipe process. The following table lists the values and default for this field.

Item	Description
Values	A numeric value between 1 and 4095. **Note**: A value can be set for this field only if the Deallocation Synchronization option is set with a value of Y.
Default	None

Writer Skip Alloc Synch

In this field, you can specify whether the writer job, that might otherwise wait during allocation for allocation synchronization, should be excluded from the allocation synchronization process. For more information on the Allocation Synchronization option, see “Allocation Synchronization” on page 9-35. The following table lists the values and default for this field.

Item	Description
Values	This option has the following possible values: • Y–The writer job is excluded from the allocation synchronization process. The writer is counted toward the number of writers required for allocation synchronization but does not wait during allocation. • N–The writer is not excluded from the allocation synchronization process. If allocation synchronization is requested, the writer will wait during allocation until all required participants have allocated the pipe. **Note**: A value of Y can be set for this field only if the Allocation Synchronization option is set with a value of Y.
Default	The value specified in JOPDEFRL.

Writer Skip Open Synch

In this field, you can specify whether the writer job, that might otherwise wait during open for open synchronization, should be excluded from the open synchronization. For more information on the Open Synchronization option, see “Open Synchronization” on page 9-35. The following table lists the values and default for this field.

Item	Description
Values	This option has the following possible values: • Y–The writer job is excluded from the open synchronization process. The writer is counted toward the number of writers required for open synchronization but does not wait during open. • N–The writer job is not excluded from the open synchronization process. If open synchronization is requested, the writer will wait during open until all the required participants open the pipe. **Note**: A value of Y can be set for this field only if the Open Synchronization option is set with a value of Y.
Default	The value specified in JOPDEFRL.

Reader Options

The Reader options allow you to specify the following values:

- Minimum number of readers necessary for synchronization
- Number of additional readers that can join the pipe
- How data is assigned to the readers
- Reader error option
- Reader selection criteria
  - jobname

- — step name
- — procedure step name
- — DD name
- Reader participant options
  - — acceptable job step condition codes
  - — End-Of-File indication processing
  - — participant specific allocation/open synchronization processing

Minimum Readers

In this field, you can specify the minimum number of readers that must reach each synchronization point before synchronization is satisfied. The following table lists the value and default for this field.

Item	Description
Values	A numeric value between 1 and 99.
Default	1

Additional Readers

In this field, you can specify the number of additional readers that can join the pipe. The sum of the values specified in the Minimum readers field and this field is the maximum number of readers that can read data from the pipe. The following table lists the value and default for this field.

Item	Description
Values	This option has the following possible values: • Y–Job Optimizer Pipes will allow any number of additional readers to join the pipe. • N–Job Optimizer Pipes will not allow additional readers to join the pipe. • A numeric value between 1 and 99–Job Optimizer Pipes will allow the specified number of additional readers to join the pipe. **Note**: A value of N must be set for this field if the Reader Data option is set to a value of A.
Default	N

Reader Data

In this field, you can indicate whether data in the pipe is read by all readers or only by the next available reader. The following table lists the values and default for this field.

Item	Description
Values	This option has the following possible values: • A–All data in the pipe is read by all reader jobs. • N–The next available reader reads the next block. Another reader does not read data already read. **Note**: A value of A can be set for this option only if the Minimum Readers option is set with a value greater than 1, the Additional Readers option is set with a value of N and the Open Synchronization option is set with a value of Y.
Default	N

Reader Error Option

In this field, you can specify how Job Optimizer Pipes handles reader jobs when it encounters an error in the pipe process or failure in a pipe participant. The following table lists the values and default for this field.

Item	Description
Values	This option has the following possible values: • I–Immediate. Job Optimizer Pipes abends all reader jobs immediately when an error occurs. • D–Delay. When an error occurs, all readers, except the reader who experienced the error (if any), continue to access the pipe. When all data from the pipe has been read, Job Optimizer Pipes abends these readers. • G–Ignore. When an error occurs, all readers, except the reader who experienced the error (if any), continue to access the pipe. When all data from the pipe has been read, these readers end normally.
Default	The value specified in JOPDEFRL.

Reader Jobs

In this field, you can specify the name of the job that reads data from the pipe data set. This field is used as selection criteria. The following table lists the values and default for this field.

Item	Description
Values	A 1- to 8-character value that adheres to JCL job name standards. Mask characters can be used.
Default	*

Reader Step Name

In this field you can specify the name of the step that reads data from the pipe data set. This parameter is optional. When specified, this field is used as selection criteria. The following table lists the values and default for this field.

Item	Description
Values	A 1- to 8-character value that adheres the JCL standards for step names. This field supports character masking.
Default	None

Reader Procedure Step Name

In this field you can specify the step name that invokes the procedure that reads data from the pipe data set. This parameter is optional. When specified, this field is used as selection criteria. The following table lists the values and default for this field.

Item	Description
Values	A 1- to 8-character value that adheres the JCL standards for step names. This field supports character masking.
Default	None

Reader DD Name

In this field you can specify the name of the data definition (DD) statement that is used to access the pipe data set. This parameter is optional. When specified, this field is used as selection criteria. The following table lists the values and default for this field.

Item	Description
Values	A 1- to 8-character value that adheres the JCL standards for DD names. This field supports character masking.
Default	None

Reader Waits for EOF

In this field, you can specify whether the specified reader job must receive an End-Of-File (EOF) indication before closing the data set. The following table lists the values and default for this field.

Item	Description
Values	This option has the following possible values: • Y–The reader job should receive an EOF indication before closing the data set. • N–The reader job can close the data set without receiving an EOF indication. **Note**: If this option is set to the Y value, the Writer Signals EOF option should be set to C or D for at least one writer.
Default	The value specified in JOPDEFRL.

Reader Error Condition Code

In this field, you can specify the lowest step condition code that Job Optimizer Pipes will treat as an error. This parameter is optional. When specified, Job Optimizer Pipes will use the step completion code to determine if there is an error in the pipe process. If the step that is reading from the pipe ends with a completion code equal to or higher than the value set for this field, Job Optimizer Pipes treats this as an error in the pipe process. The following table lists the values and default for this field.

Item	Description
Values	A numeric value between 1 and 4095. **Note**: A value can be set for this field only if the Deallocation Synchronization option is set with a value of Y.
Default	None

Reader Skip Alloc Synch

In this field, you can specify whether the reader job, that might otherwise wait during allocation for allocation synchronization, should be excluded from the allocation synchronization process. For more information on the Allocation Synchronization option, see “Allocation Synchronization” on page 9-35. The following table lists the values and default for this field.

Item	Description
Values	This option has the following possible values: • Y–The reader job is excluded from the allocation synchronization process. The reader is counted toward the number of readers required for allocation synchronization but does not wait during allocation. • N–The reader is not excluded from the allocation synchronization process. If allocation synchronization is requested, the reader will wait during allocation until all required participants have allocated the pipe. **Note**: A value of Y can be set for this field only if the Allocation Synchronization option is set with a value of Y.
Default	The value specified in JOPDEFRL.

Reader Skip Open Synch

In this field, you can specify whether the reader job, that might otherwise wait during open for open synchronization, should be excluded from the open synchronization. For more information on the Open Synchronization option, see "Open Synchronization" on page 9-35. The following table lists the values and default for this field.

Item	Description
Values	This option has the following possible values: • Y–The reader job is excluded from the open synchronization process. The reader is counted toward the number of readers required for open synchronization but does not wait during open. • N–The reader job is not excluded from the open synchronization process. If open synchronization is requested, the reader will wait during open until all the required participants open the pipe. **Note**: A value of Y can be set for this field only if the Open Synchronization option is set with a value of Y.
Default	The value specified in JOPDEFRL.

Chapter 10 Defining Job Optimizer Pipes Initialization Parameters

This chapter describes Job Optimizer Pipes Initialization parameters. This includes information about the panels and dialog actions used to create, view, or modify the initialization parameters online. This chapter discusses the following topics:

Understanding the Initialization Parameters

Job Optimizer Pipes initialization parameters define the start-up parameters for Job Optimizer Pipes. The parameters reside in a partitioned data set member. During initialization, Job Optimizer Pipes reads this member to establish its operational environment. During execution, Job Optimizer Pipes can be instructed to read this member and update its operational environment according to changes done to the parameters.

Job Optimizer Pipes uses the BatchPlex member PIPEPARMS parameter to determine which initialization parameters member to read.

Registering a Initialization Parameters Member

This section describes how to use the MAINVIEW® Batch Optimizer dialog to perform Job Optimizer Pipes initialization parameters member registration tasks.

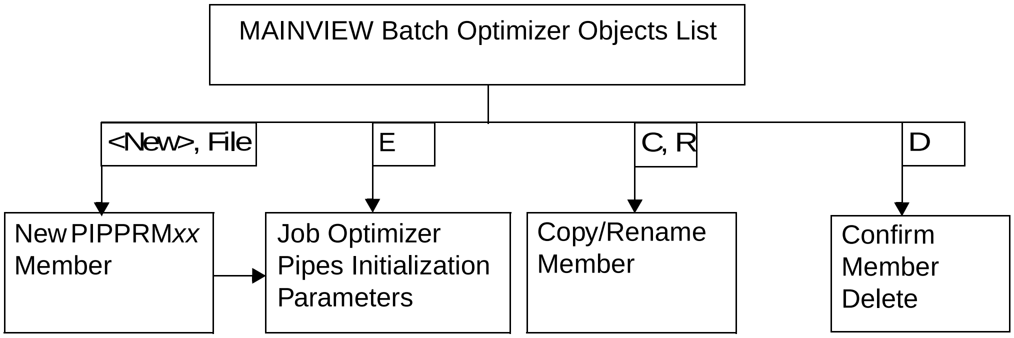
The user interface includes general help screens for each panel. It also provides input field help to describe operational considerations and valid field values. For more information about how to access the MAINVIEW Batch Optimizer dialog and online Help, see Appendix B, “Navigating the User Interface.”

The default name of the Job Optimizer Pipes initialization parameters member is PIPPRM*xx*. This name is used in this section when referring to the Job Optimizer Pipes initialization parameters member.

Panel Flow for a Job Optimizer Pipes Initialization Parameters

Figure 10-1 shows the panel flow when using the MAINVIEW Batch Optimizer dialog to edit the Job Optimizer Pipes initialization parameters member (PIPPRM*xx*). This is the basic panel flow indicating the action codes (E, C, R, D) that you select to move from one panel to another.

Figure 10-1 PIPPRM*xx* Registration Panel Flow


MAINVIEW Batch Optimizer Objects List
<New>, File
E
C, R
D
New PIPPRMxx Member
Job Optimizer Pipes Initialization Parameters
Copy/Rename Member
Confirm Member Delete

Note: For more information about creating a new Job optimizer Pipes initialization parameters member, see Appendix B, “Navigating the User Interface.”

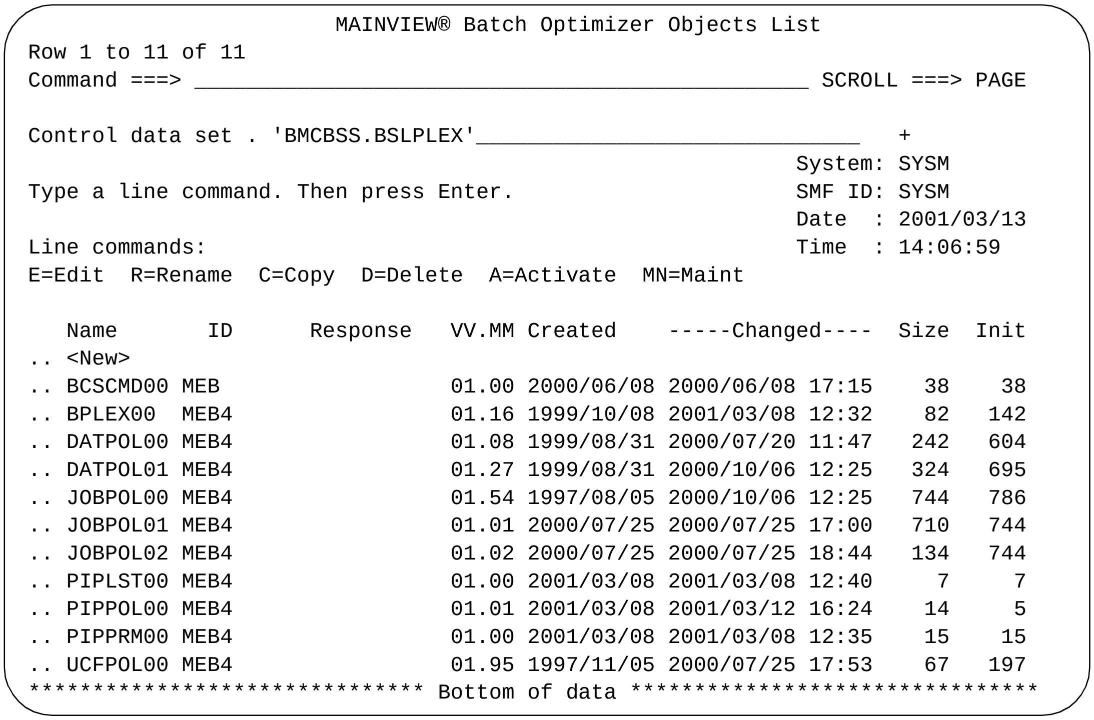
Editing or Viewing a Job Optimizer Pipes Initialization Parameters

You can edit or view the information in a Job Optimizer Pipes initialization parameters member.

To edit or view information for a Job Optimizer Pipes initialization parameters member in the control data set, complete the following steps:

Step 1 Access the MAINVIEW Batch Optimizer Objects List panel (see Figure 10-2).

Figure 10-2 MAINVIEW Batch Optimizer Objects List

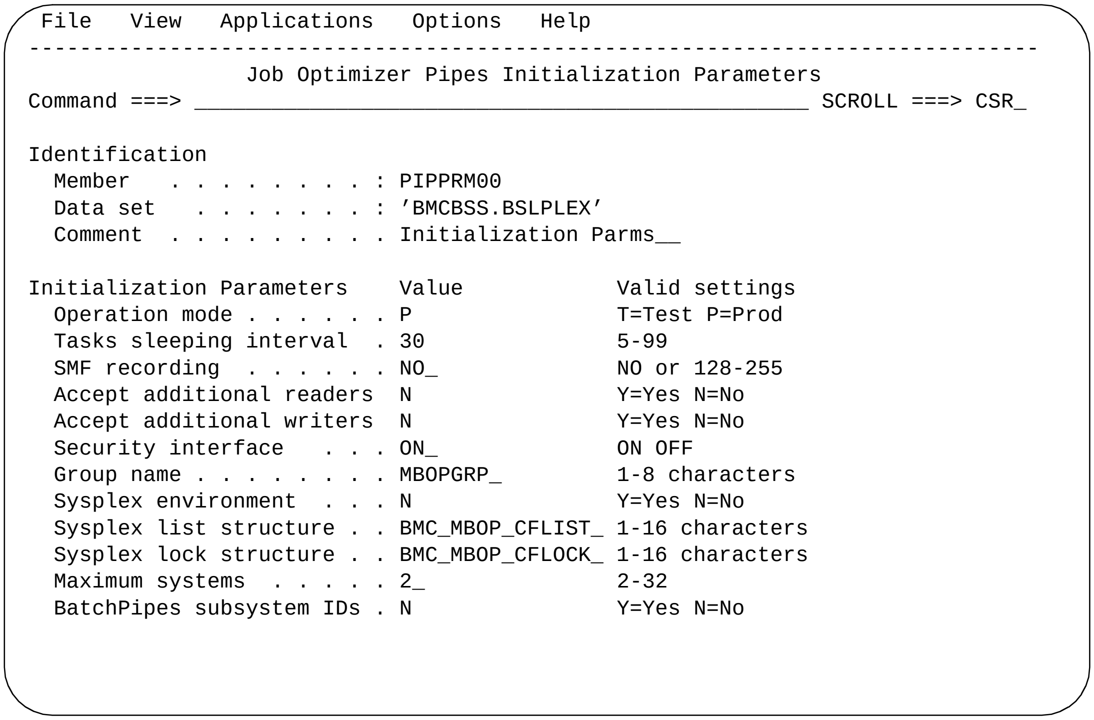


```
                        MAINVIEW® Batch Optimizer Objects List
Row 1 to 11 of 11
Command ===> ________________________________________________ SCROLL ===> PAGE

Control data set . 'BMCBSS.BSLPLEX'______________________________   +
                                                            System: SYSM
Type a line command. Then press Enter.                      SMF ID: SYSM
                                                            Date  : 2001/03/13
Line commands:                                              Time  : 14:06:59
E=Edit  R=Rename  C=Copy  D=Delete  A=Activate  MN=Maint

   Name       ID      Response   VV.MM Created    -----Changed----  Size  Init
.. <New>
.. BCSCMD00 MEB                  01.00 2000/06/08 2000/06/08 17:15    38    38
.. BPLEX00  MEB4                 01.16 1999/10/08 2001/03/08 12:32    82   142
.. DATPOL00 MEB4                 01.08 1999/08/31 2000/07/20 11:47   242   604
.. DATPOL01 MEB4                 01.27 1999/08/31 2000/10/06 12:25   324   695
.. JOBPOL00 MEB4                 01.54 1997/08/05 2000/10/06 12:25   744   786
.. JOBPOL01 MEB4                 01.01 2000/07/25 2000/07/25 17:00   710   744
.. JOBPOL02 MEB4                 01.02 2000/07/25 2000/07/25 18:44   134   744
.. PIPLST00 MEB4                 01.00 2001/03/08 2001/03/08 12:40     7     7
.. PIPPOL00 MEB4                 01.01 2001/03/08 2001/03/12 16:24    14     5
.. PIPPRM00 MEB4                 01.00 2001/03/08 2001/03/08 12:35    15    15
.. UCFPOL00 MEB4                 01.95 1997/11/05 2000/07/25 17:53    67   197
******************************* Bottom of data ********************************
```

Step 2 Position the cursor in the input field left of the Job Optimizer Pipes initialization parameters member (PIPPRMxx) you want to edit. Type **E** (Edit), and press **Enter**.

The Job Optimizer Pipes Initialization Parameters panel is displayed (Figure 10-3). This panel lists the Job Optimizer Pipes initialization parameters. You created a sample PIPPRM*xx* member during the customization process in preparation to execute the installation verification jobs. Customize it to the needs of your site.

Figure 10-3 Job Optimizer Pipes Initialization Parameters Panel



```
 File   View   Applications   Options   Help
-------------------------------------------------------------------------------
                 Job Optimizer Pipes Initialization Parameters
Command ===> ________________________________________________ SCROLL ===> CSR_

Identification
  Member   . . . . . . . . : PIPPRM00
  Data set   . . . . . . . : 'BMCBSS.BSLPLEX'
  Comment  . . . . . . . . . Initialization Parms__

Initialization Parameters    Value            Valid settings
  Operation mode . . . . . . P                T=Test P=Prod
  Tasks sleeping interval  . 30               5-99
  SMF recording  . . . . . . NO_              NO or 128-255
  Accept additional readers  N                Y=Yes N=No
  Accept additional writers  N                Y=Yes N=No
  Security interface   . . . ON_              ON OFF
  Group name . . . . . . . . MBOPGRP_         1-8 characters
  Sysplex environment  . . . N                Y=Yes N=No
  Sysplex list structure . . BMC_MBOP_CFLIST_ 1-16 characters
  Sysplex lock structure . . BMC_MBOP_CFLOCK_ 1-16 characters
  Maximum systems  . . . . . 2_               2-32
  BatchPipes subsystem IDs . N                Y=Yes N=No
```

Step 3 To make changes to any entry field that is listed on this panel, position the cursor to the field, clear the field, then type in the new text.

Step 4 Press **Enter** to save the changes or press PF12 (Cancel) to discard your changes.

Step 5 You can commit your changes to the control data set by using one of the following methods:

- Type **Save** on the Command Line, and press **Enter**. When the change is saved, the Job Optimizer Pipes initalization parameters panel is displayed again.

- Select the File pull-down choice 2 (Save...) When the change is saved, the Job Optimizer Pipes initialization parameters panel is displayed.

- Press **F12** (Cancel). The Confirm Cancel pop-up panel is displayed. Type **1** (Save changes to disk and cancel), and press **Enter**. The dialog saves the changed Job Optimizer Pipes initialization parameters member and re-displays the MAINVIEW Batch Optimizer Objects List panel.

- Press **F3** (Exit). The Confirm Exit pop-up is displayed. Type **1** (Save Changes to disk and exit), and press **Enter**. The dialog saves the changed Job Optimizer Pipes initialization parameters member and displays the MAINVIEW Batch Optimizer Objects List panel.

Activating Changes to Job Optimizer Pipes Initialization Parameters

Changes that are made to the initialization parameters can be dynamically activated by using the RELOAD command. For more information about this command, see “Reloading the Job Optimizer Pipes Initialization Parameters” on page 11-8. Not all the parameters can be dynamically changed. To determine whether the parameter can be dynamically changed or a restart of Job Optimizer Pipes is required for the change to take effect, see the “Dynamic” field in each parameter description.

Job Optimizer Pipes Initialization Parameters

Job Optimizer Pipes reads the initialization parameters member to establish or update its operational environment. The section describes the purpose of each initialization parameter and whether the parameter can be dynamically changed. The parameters are grouped into the following sections:

- identification of the control data set and member
- operational parameters

Identification

The Identification section lists the control data set name and Job Optimizer Pipes initialization parameters member that are being edited. You can provide a 22-character comment that, if included, will be placed after the PIPEPARMS keyword in the pipes initialization parameter member.

Operational Parameters

This section identifies the initialization parameters for Job Optimizer Pipes.

Operation Mode

In this field, specify the Job Optimizer Pipes operation mode. Indicate whether Job Optimizer Pipes should operate in Production mode (P) or Test mode (T). The following table lists the values, default, dynamic, and keyword for this field.

Item	Description
Values	This option has the following possible values: • P–Job Optimizer Pipes operates in production mode. In production mode, Job Optimizer Pipes operates as normal, replacing data sets that are matched by rules with pipes. • T–Job Optimizer Pipes operates in test mode. In test mode, Job Optimizer Pipes does not intervene in job processing, but instead, issues messages to the job log to indicate that these jobs are eligible for pipe processing. **Note**: Job Optimizer Pipes always handles pipes defined using the JCL SUBSYS= keyword, regardless of the operation mode.
Default	P
Dynamic	YES
Keyword	OPERMODE={PROD|TEST}

Tasks Sleeping Interval

In the Tasks Sleeping Interval field, specify the sleeping interval of the Job Optimizer Pipes address space. The Job Optimizer Pipes address space wakes periodically at pre-defined intervals, checks for timeout events, and performs housekeeping tasks when required. After performing any outstanding tasks, the address space becomes dormant until the next occurrence of the specified interval. The following table lists the values, default, dynamic, and keyword for this field.

Item	Description
Values	A numeric value, between 5 and 99 - the value specifies the number of seconds for the sleeping interval. The value is multiplied by 100 (to represent the number of hundredths of seconds) before it is stored in the parameter member.
Default	30
Dynamic	YES. The Tasks Sleeping Interval can also be dynamically changed without updating the initialization parameters. (See “Modifying the Job Optimizer Pipes Sleeping Interval” on page 11-7.) Using this method, the change will only effect the current execution of Job Optimizer Pipes address space.
Keyword	INTERVAL={500-9900}

SMF Recording

In the SMF Recording field, specify whether Job Optimizer Pipes should generate SMF records containing pipe and participants statistics. An SMF record is written when:

- A participant closes the pipe. The record contains participant I/O information.
- All the participants close the pipe. The record contains pipe I/O information and I/O information about each pipe participant.

The following table lists the values, default, dynamic, and keyword for this field.

Item	Description
Values	This option has the following possible values: • A numeric value between 128 and 255 - The SMF record ID Job Optimizer Pipes will use when generating SMF records. • NO–Job Optimizer Pipes will not generate SMF records during pipes processing.
Default	NO
Dynamic	YES
Keyword	SMFREC={NO|128-255}

Accept Additional Readers

In the Accept Additional Readers field, specify whether Job Optimizer Pipes will always allow additional readers to join a BatchPipes-compatible pipe. This parameter applies for BatchPipes-compatible pipes only. For more information, see Appendix E, “Migrating from BatchPipes to Job Optimizer Pipes.”

Warning! Allowing additional readers to join the pipe will cause significant performance degradation. Therefore, the default should not be changed unless absolutely necessary. If multiple readers are required for specific pipes, their subsystem parameters should be adjusted (ALLOCSYNC, ALLOCNOW, OPENSYNC, OPENNOW) or the pipe can be converted to be a Job Optimizer Pipes pipe (using a rule).

The following table lists the values, default, dynamic, and keyword for this field.

Item	Description
Values	This option has the following possible values: • Y–Job Optimizer Pipes allows up to 99 additional readers to join a BatchPipes-compatible pipe. • N–Job Optimizer Pipes does not allow additional readers to join a BatchPipes-compatible pipe.
Default	N
Dynamic	YES
Keyword	ADD-RDRS={YES|NO}

Accept Additional Writers

In the Accept Additional Writers field, specify whether Job Optimizer Pipes will always allow additional writers to join a BatchPipes-compatible pipe. This parameter applies for BatchPipes-compatible pipes only. For more information, See Appendix E, “Migrating from BatchPipes to Job Optimizer Pipes.”

Warning! Allowing additional writers to join the pipe will cause significant performance degradation. Therefore, the default should not be changed unless absolutely necessary. If multiple writers are required for specific pipes, their subsystem parameters should be adjusted (ALLOCSYNC, ALLOCNOW, OPENSYNC, OPENNOW) or the pipe can be converted to be a Job Optimzer Pipes pipe (using a rule).

The following table lists the values, default, dynamic, and keyword for this field.

Item	Description
Values	This option has the following possible values: • Y–Job Optimizer Pipes allows up to 99 additional writers to join a BatchPipes-compatible pipe. • N–Job Optimizer Pipes does not allow additional writers to join a BatchPipes-compatible pipe.
Default	N
Dynamic	YES
Keyword	ADD-WTRS={YES|NO}

Security Interface

In the Security Interface field, specify whether Job Optimizer Pipes should activate its security interface. The following table lists the values, default, dynamic, and keyword for this field.

Item	Description
Values	This option has the following possible values: • ON–Job Optimizer Pipes requires that a pipe participant have authority to access the pipe. When a participant opens a pipe, the security interface verifies that the participant has authorization to access the data set replaced by the pipe. Reader participants must have READ access to the data set and Writer participants must have UPDATE access to the data set. • OFF–Job Optimizer Pipes does not require that a pipe participant have authority to access the pipe.
Default	ON
Dynamic	YES
Keyword	SECURITY={ON|OFF}

Group Name

This parameter is only used by IOA users wishing to implement the interface between Control-M and MAINVIEW Batch Optimizer. If you are not an IOA user, keep the default value. For more information, see Appendix F, “IOA and INCONTROL User Considerations.” The following table lists the default, values, default, dynamic, and keyword for this field.

Item	Description
Values	1-8 character name
Default	MBOPGRP
Dynamic	NO
Keyword	GRPNAME={1-8 character name}

Sysplex Environment

In the Sysplex Environment field, specify whether Job Optimizer Pipes operates in Global mode in a Parallel Sysplex environment. For more information on Job Optimizer Pipes and sysplex environment, see *MAINVIEW Batch Optimizer Installation Manual*. The following table lists the values, default, dynamic, and keyword for this field.

Item	Description
Values	This option has the following possible values: • Y–Job Optimizer Pipes works in Global mode in a Parallel Sysplex environment. It will handle pipes between applications that run on different systems. When working in Global mode, Job Optimizer Pipes uses standard Parallel Sysplex services. • N–Job Optimizer Pipes works in a non-Parallel Sysplex environment or works in a Parallel Sysplex environment in Local mode, managing each system separately. Job Optimizer Pipes can only handle pipes between applications that run on the same system.
Default	N
Dynamic	NO
Keyword	SYSPENV={YES|NO}

Sysplex List Structure

In the Sysplex List Structure field, specify the Parallel Sysplex List structure name, which will be used for communication between the Job Optimizer Pipes address spaces working in Global mode in a Parallel Sysplex environment. To facilitate Job Optimizer Pipes' use of Parallel Sysplex services, define the necessary sysplex structures as described in the *MAINVIEW Batch Optimizer Installation Manual*. This parameter is required only if Y is set in the Sysplex Environment field. Otherwise this parameter is ignored. The following table lists the values, default, dynamic, and keyword for this field.

Item	Description
Values	1-16 character name that can contain numeric characters, uppercase alphabetic characters, national characters (@#$) and underscores. The value must start with an uppercase character.
Default	BMC_MBOP_CFLIST
Dynamic	NO
Keyword	SYSPLST={1-16 character name}

Sysplex Lock Structure

In the Sysplex Lock Structure field, specify the Parallel Sysplex Lock structure name, which will be used as a locking mechanism between the Job Optimizer Pipes address spaces working in Global mode in a Parallel Sysplex environment. To facilitate Job Optimizer Pipes' use of Parallel Sysplex services, define the necessary Sysplex structures as described in the *MAINVIEW Batch Optimizer Installation Manual*. This parameter is required only if Y is set in the Sysplex Environment field. Otherwise this parameter is ignored. The following table lists the values, default, dynamic, and keyword for this field.

Item	Description
Values	1-16 character name that can contain numeric characters, uppercase alphabetic characters, national characters (@#$) and underscores. The value must start with an uppercase character.
Default	BMC_MBOP_CFLOCK
Dynamic	NO
Keyword	SYSPLCK={1-16 character name}

Maximum Systems

In the Maximum Systems field, specify the maximum number of MVS images where Job Optimizer Pipes will be active in Global mode. When increasing the value of this parameter, recalculate the Job Optimizer Pipes Parallel Sysplex List structure size and redefine it. See *MAINVIEW Batch Optimizer Installation Manual* for information on how to accomplish this. This parameter is required only if Y is set in the Sysplex Environment field. Otherwise this parameter is ignored. This table lists the values, default, dynamic, and keyword for this field.

Item	Description
Values	A numeric value between 2 and 32 that identifies the maximum number of images.
Default	2
Dynamic	NO
Keyword	MAXSYS#={2-32}

BatchPipes Subsystem IDs

In the BatchPipes Subsystem IDs field, specify whether you want to implement Phased Migration from BatchPipes to Job Optimizer Pipes.When Phased Migration is implemented, Job Optimizer Pipes will dynamically redirect jobs using BatchPipes pipes to use Job Optimizer Pipes. For more information, see Appendix E, “Migrating from BatchPipes to Job Optimizer Pipes.” The following table lists the values, default, dynamic, and keyword for this field.

Item	Description
Values	This option has the following possible values: • Y–Implement Phased Migration. When Y is specified a pop-up panel is displayed. In the pop-up panel specify the names of the BatchPipes subsystems that are subject to phased migration. Up to 10 subsystem names can be defined. Each subsystem name is 1-4 characters. • N–Do not implement Phased Migration.
Default	N
Dynamic	YES
Keyword	BPSUBSYS=”{up to 10 subsystem names, 1-4 characters each separated by comas}”

BMC Software, Inc., Confidential and Proprietary Information

BMC Software, Inc., Confidential and Proprietary Information

Chapter 11 Job Optimizer Pipes Operator Commands

This chapter describes the operator commands that are provided by Job Optimizer Pipes for controlling its process and displaying pipe, participant, and rule-related information. The chapter also includes the display commands output formats. This chapter discusses the following topics:

Command Scope

Command Scope defines which Job Optimizer Pipes address spaces will be affected by the operator commands.

When Job Optimizer Pipes operates in Local mode, each system is handled independently. A command that is issued for a Job Optimizer Pipes address space affects only this address space.

When Job Optimizer Pipes operates in Global mode in a sysplex environment, a command may affect only the specific Job Optimizer Pipes address space in the system where the command was issued (as in Local Mode) or it can be distributed to all the Job Optimizer Pipes address spaces working in Global Mode in the sysplex. The SCOPE command parameter tells Job Optimizer Pipes how to handle the command.

Note: Several commands do not support the SCOPE parameter. These commands affect only the Job Optimizer Pipes address space for which the command was issued.

Table 11-1 shows the parameters and values of the SCOPE parameter.

Table 11-1 SCOPE Parameter Values

Value	Description
GLOBAL	Distribute the command to all Job Optimizer Pipes address spaces working in Global mode in all systems. GLOBAL is the default for all commands supporting the SCOPE parameter except SHUT. The SHUT command is executed only in the system where it was issued, unless SCOPE=GLOBAL is specified.
LOCAL	Perform the command only in the specific Job Optimizer Pipes address space in the system where the command was issued.

Job Optimizer Pipes Control Commands

This section describes the Job Optimizer Pipes control commands.

Activating Job Optimizer Pipes

Job Optimizer Pipes services are provided by the Job Optimizer Pipes address space. A separate address space runs on each system in which Job Optimizer Pipes services are required. The address space name is equal to the Job Optimizer Pipes subsystem name that is defined in BatchPlex parameters member parameter BPSSIDR. Job Optimizer Pipes is usually started by the MAINVIEW Batch Optimizer address space as part of its initialization process. To activate Job Optimizer Pipes manually, issue the following command:

```
S MBOP,SUB=MSTR | JESx
```

Use the SUB=MSTR parameter if starting Job Optimizer Pipes when JES is not up. Use the SUB=JESx parameter if starting Job Optimizer Pipes when JES is up. Specify the name of the primary subsystem (JES2, JES3, other).

When Job Optimizer Pipes is successfully activated, the following message is displayed on the console:

```
BMC282106I JOB OPTIMIZER PIPES IS ACTIVE. NAME=MBOP ID jobid
```

When the address space is active, if you try to activate an additional Job Optimizer Pipes address space with the same subsystem name on the same system, the new (that is, additional) address space immediately shuts down and an appropriate message is issued.

Shutting Down Job Optimizer Pipes

If it is necessary to shut down Job Optimizer Pipes, issue the following command:

```
F MBOP,SHUT TYPE=NORMAL|FORCE[,SCOPE=LOCAL|GLOBAL]
```

Table 11-2 shows the parameters and values of the specified command.

Table 11-2 TYPE Parameter Values

Value	Description
NORMAL	Requests that Job Optimizer Pipes perform normal termination. During normal termination, Job Optimizer Pipes first switches to Draining Mode. Under that operation mode, all active jobs/applications using Job Optimizer Pipes' services are allowed to terminate their processes, but no new jobs/applications are accepted and no new pipes are defined. When there are no more active jobs/applications, Job Optimizer Pipes terminates. While operating in this mode, you can perform other commands (e.g., Display), request forced termination, or cancel the shutdown request and resume standard operation.
FORCE	Forces Job Optimizer Pipes to terminate immediately, causing abends to all the participants which are currently using its services.

For information about the SCOPE parameter, see "Command Scope" on page 11-3.

After a few seconds, the Job Optimizer Pipes address space starts its termination process and the following message is displayed:

```
BMC282113I SHUT DOWN UPON REQUEST FROM OPERATOR
```

If normal termination was requested and there are jobs/applications using Job Optimizer Pipes services, Job Optimizer Pipes displays the active pipes/participants and issues the following message:

```
BMC282120A TO CONTINUE TERMINATION, REPLY ’R’- RETRY,
’A’- ABORT, ’W’- WAIT, OR ’F’- FORCE
```

You can cancel the shutdown request (A), force termination (F), wait until all participants terminate (W), or manually cancel the participants and tell Job Optimizer Pipes to retry termination (R).

When the termination is concluded, the following message is displayed on the console:

```
BMC282103I JOB OPTIMIZER PIPES SHUTTING DOWN
```

Note: The command P MBOP should not be used. If it is used, it will cause normal termination of the specific Job Optimizer Pipes address space, but will not allow any further operator commands during the termination process.

In an emergency, the Job Optimizer Pipes address space can be canceled. However, this action is not recommended because it may not release all resources (for example, CSA storage) held by the Job Optimizer Pipes address space.

Modifying the Job Optimizer Pipes Operation Mode

Job Optimizer Pipes receives new applications requiring its services and provides services to applications which are already connected to it. It may occasionally become necessary or desirable to temporarily prevent new applications from requesting services from Job Optimizer Pipes while continuing to provide services to applications already connected to it, and then afterwards resume normal processing. This can be done using the following command:

```
F MBOP,SET MODE=DRAIN|RESUME[,SCOPE=LOCAL|GLOBAL]
```

Table 11-3 shows the parameters and values of the specified command.

Table 11-3 MODE Parameter Values

Value	Description
DRAIN	Job Optimizer Pipes works in DRAINING mode. In this mode, Job Optimizer Pipes does not accept new participants, but existing participants can continue processing until normal termination.
RESUME	Job Optimizer Pipes resumes normal operation (Active) mode.

For more information on the SCOPE parameter, see “Command Scope” on page 11-3.

After the command is executed, the following message is displayed on the console:

```
BMC282101I JOB OPTIMIZER PIPES OPERATES IN mode MODE
```

The *mode* value is either DRAINING or ACTIVE, depending on the command issued.

Example

The following command changes the operation mode of all Job Optimizer Pipes address spaces in the sysplex to DRAINING:

```
F MBOP,SET MODE=DRAIN,SCOPE=GLOBAL
```

Modifying the Job Optimizer Pipes Sleeping Interval

Job Optimizer Pipes wakes periodically at a predefined interval, checks for time-out events and performs housekeeping tasks when required. This time interval is defined in the Job Optimizer Pipes initialization parameters, which can be changed by the system administrator and reloaded. In addition, during execution, you can modify the sleeping interval by issuing a modify command. For more information about the Job Optimizer Pipes sleeping interval see “Tasks Sleeping Interval” on page 10-7.

To modify the Job Optimizer Pipes sleeping interval during Job Optimizer Pipes execution, issue the following command:

```
F MBOP,SET INTERVAL=nn[,SCOPE=LOCAL|GLOBAL]
```

Table 11-4 shows the parameters and values of the specified command.

Table 11-4 Modifying the Sleeping Interval Parameter and Value

Parameter	Value	Description
INTERVAL	A numeric value between 5 and 99	New sleeping interval, in seconds.

For information about the SCOPE parameter, see “Command Scope” on page 11-3.

When the modification is accepted by Job Optimizer Pipes, the following message is displayed on the console:

```
BMC282144I JOB OPTIMIZER PIPES INTERVAL IS SET TO ###
SECONDS
```

The modification affects the current execution of Job Optimizer Pipes.

Example

The following command sets the sleeping interval of Job Optimizer Pipes to 20 seconds:

```
F MBOP,SET INTERVAL=20,SCOPE=LOCAL
```

Reloading the Job Optimizer Pipes Initialization Parameters

The Job Optimizer Pipes initialization parameters (defined in member PIPPRM*xx* pointed by BatchPlex member PIPEPARMS parameter) can be reloaded while Job Optimizer Pipes is active. For more information about the Job Optimizer Pipes initialization parameters see Chapter 10, “Defining Job Optimizer Pipes Initialization Parameters.”

To reload the Job Optimizer Pipes initialization parameters, issue the following command:

```
F MBOP,RELOAD PARM=INSTPARM[,SCOPE=LOCAL|GLOBAL]
```

For information about the SCOPE parameter, see “Command Scope” on page 11-3.

When the reload is performed by Job Optimizer Pipes, the following message is displayed on the console:

```
BMC282108I JOB OPTIMIZER PIPES RELOAD INSTPARM ENDED.
RC=rc
```

Changes to the following parameters take effect immediately following command execution:

- Operation Mode (OPERMODE)
- Tasks Sleeping Interval (INTERVAL)
- SMF Recording (SMFREC)
- Accept Additional Readers (ADD-RDRS)
- Accept Additional Writers (ADD-WTRS)
- Security Interface (SECURITY)
- BatchPipes Subsystem IDs (BPSUBSYS)

Changes to other parameters do not affect the current execution of Job Optimizer Pipes and require stopping and restarting Job Optimizer Pipes address space.

Reloading Job Optimizer Pipes Programs

When maintenance has been applied to Job Optimizer Pipes programs, a new copy of the program can be reloaded for use without having to shut down Job Optimizer Pipes for that purpose.

To reload a program, issue the following command:

```
F MBOP,RELOAD PGM=JOPxxxx[,SCOPE=LOCAL|GLOBAL]
```

The program name is provided by BMC Software.

For more information about the SCOPE parameter, see “Command Scope” on page 11-3.

When the reload is performed by Job Optimizer Pipes, the following message is displayed on the console:

```
BMC282108I JOB OPTIMIZER PIPES RELOAD JOPxxxx ENDED.
RC=rc
```

Example

The following command reloads the program JOPAIO in all Job Optimizer Pipes address spaces working in Global mode in the sysplex:

```
F MBOP,RELOAD PGM=JOPAIO,SCOPE=GLOBAL
```

Displaying Job Optimizer Pipes Address Spaces

When working in Global mode, Job Optimizer Pipes provides the ability to display information about all Job Optimizer Pipes address spaces running on different systems in the sysplex. To display the Job Optimizer Pipes address space information, issue the following command:

```
F MBOP,DISPLAY JOPPLEX=ALL
```

For an example and explanations of the display output, see “Display Job Optimizer Pipes Address Spaces Output” on page 11-30.

Rule Management

This section describes manual loading of pipe rules to the active environment, removing a pipe policy table from the active environment, holding/releasing a loaded rule, and displaying rules.

Manual Loading of Pipe Rules to the Active Environment

The Pipe Policy List (default member PIPLST00) contains a list of Pipe Policy Tables containing Pipe Rules to be loaded by Job Optimizer Pipes as it is started. To load additional tables, or to replace all the active tables or only a specific active table with a new (updated) copy of the rules in the tables, use one of the following methods:

- Enter the MAINVIEW Batch Optimizer Dialog, and use the Activate option for the Pipe Policy List (all Pipe Policy Tables) or a Pipe Policy Table (specific table).
- Issue the following command:

```
F MBOP,LOAD TABLE=table-name|ALL[,LIB=library-name][,SCOPE=LOCAL|GLOBAL]
```

Table 11-5 shows the parameters and values of the specified command.

Table 11-5 Manual Loading of Pipe Rules Parameters and Values

Parameter	Values	Description
TABLE	ALL	Use TABLE=ALL to replace all active rules with the rules definition in the Pipe Policy tables that are specified in the Pipe Policy List which is referenced by the BatchPlex member PIPELIST parameter. This includes replacing the pipe rule defaults with the contents of pipe rule defaults member (JOPDEFRL) from the control data set.
	1–8 characters Pipe Policy Table name	Use a Pipe Policy Table name to load a new table of rules or replace an existing table in memory. If the table has already been loaded by Job Optimizer Pipes, the new copy of the table replaces all the rules of the table active under the Job Optimizer Pipes.
LIB	1–44 characters Pipe Policy Table Library name	Required only when loading a specific table.
SCOPE	GLOBAL	For information on the SCOPE parameter, see “Command Scope” on page 11-3.
	LOCAL	

After the load request is received by Job Optimizer Pipes, the following message is displayed on the console:

```
BMC282263I RULE LOAD STARTED. TABLE=table|ALL
```

After the load is performed by Job Optimizer Pipes, the following message is displayed on the console:

```
BMC282264I RULE LOAD ENDED. RC=rc
```

Example

To load Pipe Policy Table ACTGPROD from ACTG.PROD.RULES to a specific Job Optimizer Pipes active environment, issue the following command:

```
F MBOP,LOAD TABLE=ACTGPROD,LIB=ACTG.PROD.RULES,SCOPE=LOCAL
```

Example

To replace all the loaded rules in all the Job Optimizer Pipes address spaces in the sysplex, with the rules specified in the Pipe Policy Tables pointed by the Pipe Policy List, issue the following command:

```
F MBOP,LOAD TABLE=ALL,SCOPE=GLOBAL
```

Removing a Pipe Policy Table from the Active Environment

During Job Optimizer Pipes execution, all rules of a Pipe Policy Table can be removed from the active environment.

To remove a loaded Pipe Policy Table from the active environment, issue the following command:

```
F MBOP,DELETE TABLE=table-name,LIB=library-name[,SCOPE=LOCAL|GLOBAL]
```

Table 11-6 shows the parameters and values of the specified command.

Table 11-6 Removing Pipe Policy Table Parameters and Values

Parameter	Values	Description
TABLE	1–8 characters Pipe Policy Table name	Name of the Pipe Policy Table to be removed from the active environment.
LIB	1–44 characters Pipe Policy Table Library name	Name of the library where the Pipe Policy Table was loaded from. Can be found by using the DISPLAY RULE=ALL command.
SCOPE	LOCAL	For information about the SCOPE parameter, see "Command Scope" on page 11-3.
	GLOBAL	

When the delete request is received by Job Optimizer Pipes, the following message is displayed on the console:

```
BMC282263I RULE DELETE STARTED. TABLE=table
```

When the delete is performed by Job Optimizer Pipes, the following message is displayed on the console:

```
BMC282264I RULE DELETE ENDED. RC=rc
```

Example

To delete all rules of Pipe Policy Table ACTGPROD loaded from library ACTG.PROD.RULES, from the active environment of all Job Optimizer Pipes address spaces active in the sysplex, issue the following command:

```
F MBOP,DELETE TABLE=ACTGPROD,LIB=ACTG.PROD.RULES,SCOPE=GLOBAL
```

Holding or Releasing a Loaded Rule

During Job Optimizer Pipes execution, rules can be excluded from the rule search process temporarily. This action is done by using the HOLD command. Held rules remain loaded but are ignored by the rule search process. A rule remains held until it is released (through the RELEASE command) or until the Pipe Policy Table containing the rule is refreshed (through the LOAD command). Holding a rule is useful when a specific run of the job (for example, restart after abend) is performed without using Job Optimizer Pipes services.

Holding or releasing rules can be performed using the rule name or pipe name:

- When using rule name, only one rule is held or released. A rule name within a Pipe Policy Table does not have to be unique. Hold and Release commands will locate the first rule with the required name in the required table according to the search order and hold or release it. Subsequent commands will locate the next rule which matches the criteria, and so on.
- When using pipe name, all rules matching the pipe name are held or released.

Holding or Releasing a Rule by Rule Name

To hold or release a specific rule by rule name, issue the following command:

```
F MBOP,HOLD|RELEASE
RULE=rule-name,TABLE=table-name,LIB=library-name
[,SCOPE=LOCAL|GLOBAL]
```

BMC Software, Inc., Confidential and Proprietary Information

Table 11-7 shows the parameters and values of the specified command.

Table 11-7 Holding or Releasing a Rule by Rule Name Parameters

Parameter	Values	Description
RULE	1-8 characters Rule name	Name of the PIpe Policy Rule to be held/released.
TABLE	1-8 characters Pipe Policy Table name	Name of the Pipe Policy Table containing the rule to be held/released. Can be found using the DISPLAY RULE=ALL command.
LIB	1-44 characters Pipe Policy Table Library name	Name of the library where the Pipe Policy Table was loaded from. Can be found using the DISPLAY RULE=ALL command.
SCOPE	LOCAL	For information on the SCOPE parameter, see "Command Scope" on page 11-3.
	GLOBAL	

When the hold or release request is received by Job Optimizer Pipes, the following message is displayed on the console:

```
BMC282263I RULE operation STARTED. RULE=rule-name
```

When the hold or release is performed by Job Optimizer Pipes, the following message is displayed on the console:

```
BMC282264I RULE operation ENDED. RC=rc
```

The *operation* value in both messages is HOLD or RELEASE, depending on the command issued.

Example

The following command holds rule ACT001 of Pipe Policy Table ACTGPROD loaded from library ACTG.PROD.RULES in a specific Job Optimizer Pipes active environment:

```
F MBOP,HOLD
RULE=ACT001,TABLE=ACTGPROD,LIB=ACTG.PROD.RULES,SCOPE
=LOCAL
```

Holding or Releasing Rule by Pipe Name

To hold or release rules by pipe name, issue the following command:

```
F MBOP,HOLD|RELEASE RULE=*,PIPE=pipe-name[,SCOPE=LOCAL|GLOBAL]
```

Table 11-8 shows the parameters and values of the specified command.

Table 11-8 Holding/Releasing Rules by Pipe Name Parameters

Parameter	Values
RULE	*
PIPE	1–44 characters pipe name
SCOPE	For more information about the SCOPE parameter, see “Command Scope” on page 11-3.

When the hold or release request is received by Job Optimizer Pipes, the following message is displayed on the console:

```
BMC282263I RULE operation STARTED. PIPE=pipe-name
```

When the hold or release is performed by Job Optimizer Pipes, the following message is displayed on the console:

```
BMC282264I RULE operation ENDED. RC=rc
```

The *operation* value in both messages is HOLD or RELEASE, depending on the command issued.

Example

The following command holds the rules defining a pipe named PROD.TRAN.FILE in all Job Optimizer Pipes address spaces in the sysplex:

F MBOP,HOLD RULE=*,PIPE=PROD.TRAN.FILE,SCOPE=GLOBAL

Displaying Rules

All the rules currently loaded in a specific Job Optimizer Pipes address space can be displayed. The display is performed according to the rules search order.

To display the loaded rules, issue the following command:

```
F MBOP,DISPLAY RULE=ALL
```

For an example and explanations of the display output, see “Display Rules Output” on page 11-31.

Pipe Control Commands

This section describes various pipe management commands.

Displaying the Pipe Information Summary

The pipe information summary contains brief information about each active pipe and its participants. To display the pipe information summary, issue the following command:

```
F MBOP,DISPLAY PIPE=ALL[,SYSTEM=sys-name]
```

Table 11-9 shows the parameters and values of the specified command.

Table 11-9 Displaying Pipe Information Summary Parameter

Parameter	Value	Description
SYSTEM	1–8 characters system name	Optional. Default: All systems. This parameter can be used when working in Global mode in a Sysplex environment to limit the scope of the display to pipes managed by the specified system (i.e., the Pipe Home System).

For an example and explanations of the display output, see “Display Pipes Information Summary Output” on page 11-32.

Example

To display the pipe information summary for active pipes and participants in all the systems in the sysplex, issue the following command:

```
F MBOP,DISPLAY PIPE=ALL
```

Displaying Specific Pipe Information

Information about a specific pipe and its participants can be displayed. The information can be requested in either of two formats, Long or Short. The Short format provides basic information about the pipe and its participants. The Long format provides detailed information used for problem determination purposes. General pipe and participant information is available until all the pipe participants disconnect from the pipe.

To display specific pipe information, issue the following command:

```
F MBOP,DISPLAY PIPE=pipe-name[,FORMAT=SHORT|LONG]
```

Table 11-10 shows the parameters and values of the specified command.

Table 11-10 Displaying Pipe Information Summary Parameters and Values

Parameter	Value	Description
PIPE	1–44 characters pipe name	Name of pipe for which to display information.
FORMAT (optional)	SHORT	basic information is displayed SHORT is the default.
	LONG	detailed information is displayed

See “Display Specific Pipe Information Output” on page 11-33 for an example and explanations of the display output.

Example

The following command displays basic information regarding pipe PROD.TRANS.FILE.

```
F MBOP,DISPLAY PIPE=PROD.TRANS.FILE
```

Forcing End-Of-File (EOF) Indication on a Pipe

If a pipe remains active after the writers have disconnected and the last writer did not write an EOF, you can terminate that pipe in an orderly fashion by forcing an external EOF indication. After a pipe is marked EOF, no data can be written to the pipe. However, data already written to the pipe can be read. This allows the readers to read the data from the pipe but prevents new writers from joining the pipe. When the readers finish reading and disconnect, the pipe is deleted.

To force an external EOF indication for a pipe, issue the following command:

```
F MBOP,FEOF PIPE=pipe-name
```

Table 11-11 shows the parameters and values of the specified command.

Table 11-11 Forcing EOF Indication on a Pipe Parameters and Values

Parameter	Value	Description
PIPE	1–44 characters pipe name	Name of pipe to force End-Of-File.

When FORCE EOF is performed by Job Optimizer Pipes, the following message is displayed:

```
BMC282146I PIPE FORCED EOF BY OPERATOR. PIPE=pipe-name ID pipe-id
```

Example

To force an EOF indication to pipe PROD.TRANS.FILE, issue the following command:

```
F MBOP,FEOF PIPE=PROD.TRANS.FILE
```

Canceling an Active Pipe

You can cancel an active pipe. When this is done, the pipe is immediately deleted and all its participants abend. Canceling a pipe may be required when one of the following situations occurs:

- A pipe remained defined with no participants, but new participants cannot join it. Canceling this pipe allows a new pipe to be defined.

- Job Optimizer Pipes is draining (after a SHUT command with TYPE=NORMAL), but active pipes are holding up the shutdown. These pipes can be forced to terminate before their participants terminate normally by canceling these pipes (or by replying F to the shutdown outstanding message).

To cancel a pipe, issue the following command:

```
F MBOP,CANCEL PIPE=pipe-name
```

Table 11-12 shows the parameters and values of the specified command.

Table 11-12 Canceling an Active Pipe Parameter and Value

Parameter	Value	Description
PIPE	1–44 characters pipe name	Name of pipe to cancel.

When the cancel is performed by Job Optimizer Pipes, the following message is displayed on the console:

```
BMC282146I PIPE CANCELLED BY OPERATOR. PIPE=pipe-name ID pipe-id
```

Example

To cancel the pipe PROD.TRANS.FILE, issue the following command:

F MBOP,CANCEL PIPE=PROD.TRANS.FILE

Unlocking a Pipe

In rare situations (for example, internal program failure, abend), a pipe might remain locked by a participant or Job Optimizer Pipes address space, preventing any access to the pipe. In this case, the pipe lock may be manually removed. Removal of the pipe's lock should be performed with the utmost care since it might expose the pipe participants to unexpected situations.

To unlock a locked pipe, issue the following command:

```
F MBOP,UNLOCK PIPE=pipe-name
```

Table 11-13 shows the parameters and values of the specified command.

Table 11-13 Unlocking a Pipe Parameter and Value

Parameter	Value	Description
PIPE	1–44 characters pipe name	Name of pipe to unlock.

When the unlock is performed by Job Optimizer Pipes, the following message is displayed on the console:

```
BMC282146I PIPE UNLOCKED BY OPERATOR. PIPE=pipe-name ID pipe-id
```

Participant Control Commands

This section describes the displaying of participant information.

Displaying Participant Information

Information regarding a participant (in one or more pipes) can be displayed using its Address Space ID. The information can be requested in two formats, Short or Long. The Short format provides basic information about the participant. The Long format provides detailed information used for problem determination purposes.

To display pipe participant information, issue the following command:

```
F MBOP,DISPLAY ASID=nnn[,FORMAT=SHORT|LONG]
```

Table 11-14 shows the parameters and values of the specified command.

Table 11-14 Displaying Participant Information Parameters and Values

Parameter	Value	Description
ASID	address space number in decimal format.	the address space in which the participant (job) is running
FORMAT (optional)	SHORT	basic information is displayed SHORT is the default.
	LONG	detailed information is displayed

For an example and explanations of the display output, see “Display Participant Information Output” on page 11-42.

Example

To display basic information about the participant (job) which runs in address space number 85, issue the following command:

F MBOP,DISPLAY ASID=85

Problem Determination Commands

Job Optimizer Pipes is supplied with internal tracing and problem determination facilities which provide you with the following abilities:

- produce an internal trace
- display statistics about the data space usage
- display Sysplex lock information
- print the contents of several Job Optimizer Pipes internal data areas
- print Coupling Facility information

Note: These facilities should be used only when requested by the BMC Software representative.

Internal Trace Management

Under normal circumstances, the tracing facilities are dormant. Internal tracing can be activated when Job Optimizer Pipes is started or during Job Optimizer Pipes execution, and it can be modified or stopped during Job Optimizer Pipes execution. The internal trace setting can be displayed. When activated, the trace information is either printed to DD statements PRTDBG and DADUMP defined in Job Optimizer Pipes started task procedure, or issued to the console (depending on the process which produces the trace information). Tracing can be requested for specific components and/or for a specific job and/or pipe. Each component has its own identifying numbers for purposes of producing the trace. These numbers are called the trace level. One or more trace levels can be activated at the same time. Trace levels can be modified during execution.

BMC Software Support can provide you with the list of the trace levels which should be used, depending on the problem encountered.

Activating/Deactivating the Internal Trace

The Internal Trace can be activated/deactivated during one of the following situations:

- when Job Optimizer Pipes is already active
- during Job Optimizer Pipes initialization

Activating or Deactivating the Internal Trace When Job Optimizer Pipes Is Active

To activate or deactivate the internal trace while Job Optimizer Pipes is active, issue the following command:

```
F MBOP,SET
DEBUG=level[,DBGJOB=jobname][,DBGPIPE=pipe-name][,SCOPE=LOCAL|GLOBAL]
```

Table 11-15 shows the parameters and values of the specified command.

Table 11-15 Activating/Deactivating the Internal Trace Parameters

Parameter	Value	Description
DEBUG	Any number between 1 and 1024, or a range of numbers (for example, 10:15), or a list of numbers and ranges between apostrophes (for example, ‘10,15,37:38’).	Trace levels that are used to activate the trace in specific components. The trace level is added to the trace levels already set.
	Any number between -1 and -1024 or a range of numbers (for example, -20:-25) or a list of numbers and ranges between apostrophes (for example,‘-10,-15,-37:-38’.)	Trace levels that are used to deactivate the trace in specific components. The number is removed from the active trace levels.
DBGJOB (optional)	1–8 characters job name	Internal tracing (according to the active trace levels) is to be performed only for the specified job.
	ALL	Internal tracing (according to the active trace levels) is to be performed for all jobs. ALL is the default.
DBGPIPE (optional)	1–44 characters pipe name	Internal tracing (according to the active trace levels) is to be performed only for the specified pipe.
	ALL	Internal tracing (according to the active trace levels) is to be performed for all pipes. ALL is the default.
SCOPE	LOCAL	For information about the SCOPE parameter, see “Command Scope” on page 11-3.
	GLOBAL	

When the trace level set is completed, all or part of the following messages appear on the console, depending on the parameters specified:

```
BMC282871I TRACE/DEBUG LEVELS SET AS FOLLOWS:
BMC282872I level:level – TURNED on/off
BMC282122I DEBUG job/pipe IS SET TO jobname/pipename
```

Example

To activate internal trace levels 20 to 29 for all the jobs and all the pipes in all the Job Optimizer Pipes address spaces in the sysplex, issue the following command:

F MBOP,SET DEBUG=20:29,SCOPE=GLOBAL

Example

To deactivate internal trace level 25 (and keep all the others) from this specific Job Optimizer Pipes address space, issue the following command:

F MBOP,SET DEBUG=-25,SCOPE=LOCAL

Example

To activate internal trace level 30 only for pipe PROD.TRANS.FILE in all the Job Optimizer Pipes address spaces in the sysplex, issue the following command:

F MBOP,SET
DEBUG=30,DBGPIPE=PROD.TRANS.FILE,SCOPE=GLOBAL

Activating the Internal Trace During Job Optimizer Pipes Initialization

If you require tracing during Job Optimizer Pipes initialization, ensure that Job Optimizer Pipes is not active, and then start a new Job Optimizer Pipes address space with the following command:

```
S MBOP,PARM='SET DEBUG=level'
```

Table 11-16 shows the parameters and values of the specified command.

Table 11-16 Activating the Internal Trace During Job Optimizer Pipes Initialization Parameter

Parameter	Value	Description
DEBUG	Any number between 1 and 1024, or a range of numbers (for example, 10:15) or a list of numbers and ranges between multiple apostrophes (for example, '10,15,37:38').	Trace levels used to activate the trace in specific components.

Stopping the Internal Trace

When you have finished with problem determination procedures, internal tracing should be stopped. Use the following command to stop internal tracing:

```
F MBOP,SET DEBUG=OFF[,SCOPE=LOCAL|GLOBAL]
```

For information on the SCOPE parameter, see "Command Scope" on page 11-3.

When tracing is stopped, the following messages are displayed on the console:

```
BMC282871I TRACE/DEBUG LEVELS SET AS FOLLOWS:
BMC282872I 0001:1024 – TURNED OFF
BMC282122I DEBUG JOB IS SET TO ALL
BMC282122I DEBUG PIPE IS SET TO ALL
```

Displaying the Internal Trace Setting

The internal trace setting (level, DBGJOB value, and DBGPIPE value) can be displayed for a specific Job Optimizer Pipes address space. Use the following command to display the internal trace setting:

```
F MBOP,SET DEBUG=SHOW
```

The following messages are displayed on the console as the result of the Show command. The format depends on the setting of the trace levels and job/pipe names:

```
BMC282873I PRESENT TRACE/DEBUG LEVELS ARE AS FOLLOWS:
BMC282874I level:level – on/off
BMC282874I level:level – on/off
BMC282122I DEBUG JOB IS SET TO name/ALL
BMC282122I DEBUG PIPE IS SET TO name/ALL
```

Displaying Data Space Information

Job Optimizer Pipes uses a data space to hold pipe buffers, pipe control blocks, and internal tables. Data space storage is logically divided into blocks called cell pools. Each cell pool is assigned a name and contains a number of cells. Job Optimizer Pipes uses different cell pools for different types of information, such as pipe buffers or messages. Some statistic information regarding cell pools utilization and data space utilization can be displayed.

To display data space information, issue the following command:

```
F MBOP,DISPLAY CLPOOL=pool-name|ALL
```

Table 11-17 shows the parameters and values of the specified command.

Table 11-17 Displaying Data Space Information Parameter and Value

Parameter	Value	Description
CLPOOL	Valid cell pool names: • BP_1K • BP_4K • BP_8K • BP_16K • BP_24K • BP_28K • BP_32K • CP_PIPB • CP_MSGB • CP_IOB	name of cell pool to display
	ALL	Display all cell pools.

Example

To displays information regarding the cell pool which contains pipe buffers for pipes with block size between 16K and 24K, issue the following command:

F MBOP,DISPLAY CLPOOL=BP_24K

Displaying Sysplex Lock Information

Job Optimizer Pipes uses Sysplex locks to serialize access within its Coupling Facility structures. In certain situations (mainly during problem determination process) it is helpful to provide information about the Sysplex locks that are held by Job Optimizer Pipes and its participants.

To display the sysplex locks held by Job Optimizer Pipes, issue the following command:

```
F MBOP,DISPLAY LOCK=ALL
```

In most cases the following message is displayed, indicating that there are no active locks:

```
BMC282188I MBOP DOES NOT HOLD ANY SYSPLEX LOCKS
```

When there are active locks, the following messages are displayed on the console, describing who holds the locks and for which pipe:

```
BMC282186I SYSTEM  JOBNAME   JOBID    ASCBADDR TCBADDR  LOCK-RID
BMC282187I OS35    PRDWTR1   JOB01720 00F89D00 009DE120 00000000000004D300000002
BMC282136I PROD.TRANS.FILE
```

These messages do not necessarily indicate a problem, as locks are held and released during normal operation. Only a lock that is held for a long period of time may indicate a problem.

Printing Internal Data

The contents of several Job Optimizer Pipes internal data areas and the contents of the data space can be printed, in dump format, to DD statement DADUMP (defined in Job Optimizer Pipes started task procedure).

BMC Software Customer Support can provide you with the type of data (and, if necessary, the list of the data area names) to be printed, depending on the problem encountered. When the print is completed, the following message is displayed on the console:

```
BMC282021I SNAP ID=snapid PRODUCED ON DADUMP DD
```

snapid is a number between 0–255, which is displayed within the printed data and identifies the requested print.

Printing Pipe-Related Tables

To print pipe-related tables, issue the following command:

```
F MBOP,SNAP PIPE=pipe-name
```

Table 11-18 shows the parameters and values of the specified command.

Table 11-18 Printing Pipe-Related Tables Parameter and Value

Parameter	Value	Description
PIPE	1–44 characters pipe name	name of pipe for which the print is requested.

Example

To print the tables related to pipe PROD.TRANS.FILE and its participants, issue the following command.

F MBOP,SNAP PIPE=PROD.TRANS.FILE

Printing Participant-Related Tables

Participant-related tables can be printed using the participant address space ID. To print the tables, issue the following command:

```
F MBOP,SNAP ASID=number
```

Table 11-19 shows the parameters and values of the specified command.

Table 11-19 Printing Participant-Related Tables Parameter and Value

Parameter	Value	Description
ASID	address space number in decimal format	the number of the address space in which the participant (job) is running.

Example

To print the tables related to the job which runs in address space ID 85, issue the following command:

F MBOP,SNAP ASID=85

Printing Data Space Contents

The contents of all or part (specific cell pools) of the data space can be printed.

To print this information, issue the following command:

```
F MBOP,SNAP CLPOOL=pool-name|ALL
```

Table 11-20 shows the parameters and values of the specified command.

Table 11-20 Printing Data Space Contents Parameter and Values

Parameter	Value	Description
CLPOOL	Valid cell pool names: • BP_1K • BP_4K • BP_8K • BP_16K • BP_24K • BP_28K • BP_32K • CP_PIPB • CP_MSGB • CP_IOB	print the contents of the requested cell pool.
	ALL	Print the contents of all cell pools.

Example

To print the contents of the cell pool containing pipe buffers for pipes with block size between 16K and 24K, issue the following command:

```
F MBOP,SNAP CLPOOL=BP_24K
```

To print the contents of the entire data space, issue the following command:

F MBOP,SNAP CLPOOL=ALL

Printing General Data Areas

To print the contents of the Job Optimizer Pipes internal data areas, issue the following command:

```
F MBOP,SNAP TABLE=table-name|ALL
```

Table 11-21 shows the parameters and values of the specified command.

Table 11-21 Printing General Data Areas Parameter and Values

Parameter	Values	Description
TABLE	Valid table names: • QJT • MMT • MTL • RQT • MNH • MLE • CDB	print the contents of the requested table.
	ALL	print the contents of all the tables listed above.

Example

To print the contents of the QJT table, issue the following command:

F MBOP,SNAP TABLE=QJT

Printing the Coupling Facility Structures Contents

The contents of the Job Optimizer Pipes Coupling Facility structures can be printed. The print is made to a SVC dump (and not to DADUMP DD, as made for all other print commands). This dump can be analyzed using IPCS.

To print the Coupling Facility Structures information, issue the following command:

```
F MBOP,DUMP TYPE=SYSPLEX
```

When the command is processed by Job Optimizer Pipes, the following messages are displayed on the console:

```
IEA794I SVC DUMP HAS CAPTURED: DUMPID=xxx REQUESTED BY JOB (MBOP)
DUMP TITLE=BMC282045I MBOP     - SYSPLEX  DUMP REQUESTED BY OPERATOR
```

BMC Software, Inc., Confidential and Proprietary Information

Display Command Output Formats

This topic contains examples of Display command outputs and their explanations. For commands having Short and Long display formats, only the Short format is explained. (Long display formats are not explained because they are usually used for problem determination, and most of the information in Long display formats is for internal use by BMC Software Customer Support.)

Note: When a block of information is displayed (for example, through DISPLAY PIPE or DISPLAY ASID), field order is always maintained. However, formatting and line breaks can change according to the data displayed.

Display Job Optimizer Pipes Address Spaces Output

The following command displays Job Optimizer Pipes address spaces active in the sysplex:

```
F MBOP,DISPLAY JOPPLEX=ALL
```

When the command is issued, the following messages are displayed:

```
BMC282174I SYSTEM   JOBNAME  JOBID    SUBSYSTEM LIST-NUM  CONNECTION-TOKEN
STATUS   VERSION
BMC282175I SYSA     MBOP     STC14792   MBOP     0006     IXCLO00A - 0001
ACTIVE     23
BMC282175I SYSB     MBOP     STC06788   MBOP     0007     IXCLO00A - 0002
ACTIVE     23
```

The display output contains a title line (BMC282174I), followed by one line (BMC282175I) for each Job Optimizer Pipes address space.

Table 11-22 shows the information that is displayed.

Table 11-22 Display Job Optimizer Pipes Address Spaces Output (Part 1 of 2)

Display	Description
SYSTEM	system name where Job Optimizer Pipes is running
JOBNAME	job name of Job Optimizer Pipes address space
JOBID	job ID (number) of Job Optimizer Pipes address space
SUBSYSTEM	subsystem name of Job Optimizer Pipes
LIST-NUM	list number (in the Coupling Facility) of this Job Optimizer Pipes address space

Table 11-22 Display Job Optimizer Pipes Address Spaces Output (Part 2 of 2)

Display	Description
CONNECTION-TOKEN	connection identifier that is assigned to the Coupling Facility connection of this Job Optimizer Pipes address space
STATUS	Status of Job Optimizer Pipes address space in that system. Possible values are: ACTIVE - pipe processing operates for existing and new pipes. INIT - Job Optimizer Pipes address space performs initialization SHUT-REQ - shutdown was requested by operator. OTHER - any other status.
VERSION	Version of Job Optimizer Pipes. Possible value is 23

Display Rules Output

The following command displays the list of loaded pipe rules:

```
F MBOP,DISPLAY RULE=ALL
```

When the command is issued, the following messages are displayed:

```
BMC282268I RULE/DSN PRIORITY MODE STATUS TABLE    LIBRARY
BMC282269I IVP               P    A      JOPIVPRL MVBO.BSLPLEX
BMC282270I MVBO.IVP.FILE
BMC282269I TRANS             P    A      PROD     MVBO.BSLPLEX
BMC282270I PROD.TRANS.FILE
BMC282269I JOPDEFRL          P    A      JOPDEFRL MVBO.BSLPLEX
BMC282270I *
```

The display output contains a title line (BMC282268I), followed by two lines for each rule displayed: BMC282269I for rule information, and BMC282270I for the dataset (pipe) name.

Table 11-23 shows the information that is displayed.

Table 11-23 Display Rule Output

Output	Description
RULE/DSN	rule name in BMC282269I. Pipe (dataset) name in BMC282270I.
PRIORITY	sequence priority that is specified in the rule (N/A)
MODE	rule operation mode (N/A)

Table 11-23 Display Rule Output

Output	Description
STATUS	rule status: • A - Active • H - Held
TABLE	name of the pipe policy table containing the rule
LIB	name of the library containing the Pipe Policy table

Display Pipes Information Summary Output

The following command displays a summary of all pipes and their participants:

```
F MBOP,DISPLAY PIPE=ALL
```

When the command is issued, the following messages are displayed:

```
BMC282148I PIPE/JOB NAME JOBID    DDNAME TYPE   SYSTEM STATUS
BMC282149I PROD.TRANS.FILE                      SYS1
BMC282150I PRCRTRNS      JOB14788 TRANS  WRITER SYS1   ACTIVE
BMC282149I ACCT.NOTES.FILE                      SYS1
BMC282150I ACTPRTNT      JOB14796 SYSUT1 READER SYS1   ACTIVE
BMC282150I ACTCRENT      JOB14795 SYSUT2 WRITER SYS1   ACTIVE
BMC282151I NUMBER OF PIPES=0002, NUMBER OF PARTICIPANTS=0003
```

The display output contains the following information:

- title line (BMC282148I)
- for each pipe displayed, one line for pipe information (BMC282149I), followed by one line for each participant connected to the pipe (BMC282150I)
- summary line (BMC282151I) containing the total number of pipes and participants

Table 11-24 shows the information that is displayed.

Table 11-24 Display Pipes Information Summary Output

Display	Description
PIPE/JOB NAME	pipe name in BMC282149I. Participant job name in BMC282150I
JOBID	participant job ID

Table 11-24 Display Pipes Information Summary Output

Display	Description
DDNAME	name of the DD statement that is used to allocate the pipe
TYPE	participant type: • READER • WRITER • UNKNWN
SYSTEM	for the pipe, the Pipe Home System (PHS) for the participant, the system in which the participant is executing
STATUS	participant status: • ACTIVE - participant is connected to the pipe • TERM - participant has disconnected from the pipe

Display Specific Pipe Information Output

The following command displays specific pipe information:

```
F MBOP,DISPLAY PIPE=PROD.TRANS.FILE
```

When the command is issued, the following messages are displayed:

```
BMC282136I PIPE: PROD.TRANS.FILE                               ID: 12585835
BMC282136I ================================================================
BMC282136I PHASE...... ACTIVE      STAT1...... ADD-RDR     STAT2......
BMC282136I CURR-WTR... 1           CURR-RDR... 0           TOT-WTR.... 1
BMC282136I TOT-RDR.... 0           LAST-ACT... 12:58:59
BMC282136I FLAG1...... DAT-PIPE,BATCH-PI                   FLAG2......
BMC282136I FLAG3......             TIMESTMP... 1258583553940000
BMC282136I HOME-SYS... OS35        SYNC-TYP... OPEN
BMC282136I RD-EROPT... IMMED       WT-EROPT... IMMED
BMC282136I I/O INFORMATION:
BMC282136I ----------------
BMC282136I BUFF#...... 9           MULTI-RD... 1           BLKS-WT.... 9
BMC282136I BLKS-RD.... 0
BMC282136I      PARTICIPANT TRNWTR
BMC282136I      ====================
BMC282136I TYPE....... WRITER      JOBNAME.... TRNWTR
BMC282136I PHASE...... IO#1DONE    WAIT-TYP... IO-WAIT     ERR-FLAG...
BMC282136I STAT1......             STAT2...... 00          FLAGS......
BMC282136I RDR/WTR.... EOF-CLOS    JOBID...... JOB09166    ASID....... 83
BMC282136I STEP-NUM... 01          PROCSTEP...
BMC282136I PGMSTEP.... S1          DDNAME..... OURFILE
BMC282136I SYSTEM..... OS35
BMC282136I I/O INFORMATION:
BMC282136I ----------------
BMC282136I BLK-CNT.... 9           LAST-TTR... 00000000    WAIT-CNT... 1
BMC282136I WAIT-TYP... IO-WAIT     LAST-I/O... 12:58:59
```

The displayed information is divided into several logical groups:

- pipe information:
  - — general pipe information (shown in Table 11-25 on page 11-35)
  - — pipe I/O information (shown in Table 11-26 on page 11-38)
- participant information, repeated for each participant that is connected to the pipe, containing for each participant:
  - — general participant information (shown in Table 11-27 on page 11-39)
  - — participant I/O information (shown in Table 11-28 on page 11-41)

General Pipe Information

Table 11-25 shows the general pipe information that is displayed.

Table 11-25 General Pipe Information (Part 1 of 3)

Pipe Information	Description
PIPE	Pipe (dataset) name.
ID	Unique pipe identifier (the first 8 characters of the pipe time-stamp).
PHASE	Current phase in the pipe life cycle: • ALOCSYNC—Pipe is not fully defined yet. Synchronization type ALLOC was requested; not all required participants performed allocation. • OPENSYNC—Pipe is not fully initialized yet. Synchronization type OPEN was requested; not all required participants opened the pipe. • WAITFWTR—Readers which already opened the pipe are waiting for the first writer to open the pipe and perform pipe initialization (synchronization at open was not requested). • INIT—Pipe initialization is in process. • ACTIVE—Pipe is active. Data can be transferred into/from the pipe. • AFTEOF—Pipe is empty and is marked with End-Of-File indication. (No more data can be read or written.) • CLOSSYNC—Some participants closed the pipe and are waiting for all the participants to close (synchronization at CLOSE was requested). • AFTCLOSE—All participants already closed the pipe. • DALCSYNC—Some participants deallocated the pipe and are waiting for all participants to deallocate (synchronization at DEALLOC was requested).

Table 11-25 General Pipe Information (Part 2 of 3)

Pipe Information	Description
STAT1	First status field. One or more of the following conditions can be indicated: • ERROR—Pipe in error. Active participants can continue process according to the Error Option that is defined in the rule. • ABEND—Pipe in error. All active participants should immediately abend (the pipe can no longer be accessed). • EOF-RCVD—End-Of-File indication was received from the last active writer. No more data can be written into the pipe. • ADD-WTR—Additional writers can still join the pipe. • ADD-RDR—Additional readers can still join the pipe.
STAT2	Second status field. The following can be indicated: • IO-ERROR—An I/O error occurred during the pipe process. • PIPE-INI—Pipe initialization is in process. • AFT-INIT—pipe initialization is completed. • STEP-END—Pipe participants are waiting during step end processing (synchronization at DEALLOC was requested).
CURR-WTR	Number of writers that are connected to the pipe (in any phase).
CURR-RDR	Number of readers that are connected to the pipe (in any phase).
TOT-WTR	Total number of writers that are and were connected to the pipe (including writers who terminated for any reason).
TOT-RDR	Total number of readers that are and were connected to the pipe (including readers who terminated for any reason).
LAST-ACT	Time of last action that was performed by any participant, and which caused a Phase or Status change (not including I/O operations).
FLAG1	First indicator field. One or more of the following conditions can be indicated: • DAT-PIPE—Pipe is used to transfer data. Default. • READ-ALL—All readers receive all the data passing through the pipe. • WTR2FILE—Writer will write all pipe data into a physical file. • OPEN-P—“Open” pipe. The pipe accepts an unlimited number of participants. • BATCH-PI—Pipe is a BatchPipes-Compatible pipe.
FLAG2	Second indicator field. One or more of the following conditions can be indicated: • GLB-PIPE—Global pipe • FAST-PIP—Global pipe with only one writer

Table 11-25 General Pipe Information (Part 3 of 3)

Pipe Information	Description
FLAG3	Third indicator field. One or more of the following can be indicated: • ERC=DUMY - BatchPipes-Compatible mode: ERC=DUMMY was specified as one of BatchPipes subsystem parameters. • STP2STP - Pipe was dynamically defined by Job Optimizer to implement parallel step processing. • NOTE-SUP - Pipe supports the NOTE service for internal checkpointing. • CLNUP-FL - All cleanup attempts failed for the pipe. Participants of this pipe may hang. • ENDLESSW - A potential endless-wait was identified for this pipe. The pipe participants will be abended.
TIMESTMP	Unique identifier (time stamp) for a pipe. The time stamp value is determined when the first participant allocates the pipe.
HOME-SYS	Name of the Pipe Home System (PHS). The system where the pipe is managed.
SYNC-TYP	Synchronization points that are defined for the pipe: • NONE (no synchronization) or any combination of the following values: • ALLOC • OPEN • CLOSE • DEALLOC
RD-EROPT	Action to be taken by a reader when an error is distributed to it from another participant: • IMMED—Abend immediately. • DELAY—Continue reading until there is nothing more to read from the pipe, and then abend. • IGNORE—Continue reading until there is nothing more to read from the pipe, and then terminate normally.
WT-EROPT	Action to be taken by a writer when an error is distributed to it from another participant: • IMMED—Abend immediately. • IGNORE—Continue writing to the file and, if possible, to the pipe. At the end of the process, terminate normally.

Pipe I/O Information

Table 11-26 shows the pipe I/O information that is displayed.

BMC Software, Inc., Confidential and Proprietary Information

Table 11-26 Pipe I/O Information

Pipe Information	Description
BUFF#	number of buffers that are allocated for this pipe
MULTI-RD	number of times that each block must be read before it can be freed This number is 1, unless there are several readers, and each reader gets all the data. In this case, it is equal to the number of readers.
BLKS-WT	total number of blocks that were written by all the writers
BLKS-RD	total number of blocks that were read by all of the readers

BMC Software, Inc., Confidential and Proprietary Information

General Participant Information

Table 11-27 shows the general participant information that is displayed.

Table 11-27 General Participant Information (Part 1 of 3)

Participant Information	Description
TYPE	participant type: • READER • WRITER
JOBNAME	participant job name
PHASE	current phase of a pipe participant: • WAITALOC—Waiting during allocation until all required participants allocated the pipe (synchronization type ALLOC was requested). • ALOCDONE—After allocation. • DEFROPEN—Open is requested and is deferred until the participant performs its first I/O (because no writer has opened the pipe yet). • WAITOPEN—Waiting during open until all required participants opened the pipe (synchronization type OPEN was requested), or waiting for pipe/connection initialization process to complete. • OPENDONE—After open. • IO#1DONE—During I/O, after the first I/O operation. • DEFRCLOS—Close was requested and is deferred until deallocation. Reopen is allowed. • EOF-CLOS—Writer has already sent EOF indication as part of its close process. • WAITCLOS—Waiting during close until all the required participants closed the pipe (synchronization type CLOSE was requested). • CLOSDONE—After close. • EOF-DLOC—Writer has already sent EOF indication as part of its deallocation process. • STEP-END—Waiting during step-end processing for the termination of all the participants of the current pipe and all the pipes of the step (Synchronization type DEALLOC was requested). • WAITDALC—Waiting during deallocation until all the required participants deallocated the pipe (synchronization type DEALLOC was requested). • DALCDONE—After deallocation. • AFT-STPE—After step-end processing.

Table 11-27 General Participant Information (Part 2 of 3)

Participant Information	Description
WAIT-TYP	If the participant is waiting for any event, this field contains the wait type (i.e., reason for waiting). • NONE—Participant is not waiting. • DCB-PARM—Reader is waiting for a Writer to open the pipe, so the reader can receive required DCB attributes (BLOCKSIZE, LRECL, RECFM) from the pipe. • ALOCSYNC—Participant is waiting for Allocation synchronization. • OPENSYNC—Participant is waiting for Open synchronization. • PIPEINIT—Participant is waiting for pipe initialization. • SPECINIT—Participant is waiting for connection initialization (it joined the pipe when the pipe was already active). • IO-WAIT—Participant is waiting for I/O (for a reader, the reader tries to read a block but the pipe is empty; for a writer, the writer tries to write a block but the pipe is full). • CLOSSYNC—Participant is waiting for Close synchronization. • STEP-END—Participant is waiting during step-end processing for the termination of all the participants of the current pipe and all the pipes of the step. • DALCSYNC—Participant is waiting for Deallocation synchronization.
ERR-FLAG	participant error flags: • REJECTED—Participant was unable to join the pipe for various reasons. • ERROR—Participant encountered an error. • DIST-ERR—An error was distributed from another participant. • IO-ERROR—Participant encountered an I/O error.
STAT1	first status field: • TERM—Participant terminated; participant will no longer access the pipe. • AFT-EOF—End-Of-File was already sent or received. • DEFROPEN—Deferred open situation. Open process to be completed before the first I/O. • RMT-PRT—A remote participant. A participant of a Global pipe which runs in a different system than the Pipe Home System.
STAT2	second status field. For future use.
FLAGS	various indications: • REOPEN—Participant reopened the pipe. • MIS-DCBP—Reader is or was waiting for a Writer to open the pipe, so the reader can receive required DCB attributes (BLOCKSIZE, LRECL, RECFM) from the pipe.

BMC Software, Inc., Confidential and Proprietary Information

Table 11-27 General Participant Information (Part 3 of 3)

Participant Information	Description
RDR/WTR	information about participants parameters: • SKP-ALCS—Participant skips ALLOC synchronization, although synchronization type ALLOC was requested for the pipe. • SKP-OPNS—Participant skips OPEN synchronization, although synchronization type OPEN was requested for the pipe. For Writer: • EOF-CLOS—Participant should send EOF indication as part of close process. • EOF-DALC—Participant should send EOF indication as part of deallocation process. For Reader: • EOF-REQ—Participant must wait for EOF before closing the pipe.
JOBID	Participant job-ID (job number).
ASID	Address space number (in decimal) in which the participant is running.
STEP-NUM	Step number, within the job, that allocates the pipe.
PROCSTEP	Name of step that invokes the current JCL procedure.
PGMSTEP	Name of step that allocates the pipe.
DDNAME	Name of DD statement used to allocate the pipe.
SYSTEM	System name where the participant is running.

Participant I/O Information

Table 11-28 shows the participant I/O information that is displayed.

Table 11-28 Participant I/O Information (Part 1 of 2)

Participant Information	Description
BLK-CNT	number of blocks written or read by the participant
LAST-TTR	when using NOTE services, shows the last written/read TTR
WAIT-CNT	number of times the participant waited for I/O

Table 11-28 Participant I/O Information (Part 2 of 2)

Participant Information	Description
WAIT-TYP	type of wait when waiting: • IO-WAIT—Waiting for I/O • NONE—Participant is not waiting. • UNKNOWN—Wait type is unknown.
LAST-ACT	time of last I/O operation

Display Participant Information Output

The following command displays participant information:

```
F MBOP,DISPLAY ASID=82
```

When the command is specified, the following messages are displayed:

```
BMC282136I          ASID 00180
BMC282136I         ============
BMC282136I JOBNAME.... E02P        JOBID...... JOB21590    STEP-NUM... 01
BMC282136I PIPE: PROD.TRANS.FILE                               ID: 11151078
BMC282136I ----------------------------------------------------------------
BMC282136I TYPE....... WRITER      PIPETYPE... MANUAL
BMC282136I LAST-ACT... 11:15:14    STAT.......
BMC282136I PHASE...... WAITOPEN    PRT-FLG....             PIPE-FLG...
BMC282136I PIPE-FL2...             STEP-NUM... 01
BMC282136I DDNAME..... OUTFILE     HOME-SYS... OS35
BMC282136I TIMESTMP... 1115107818620000                    FLAG1......
```

The displayed information is divided into the following logical groups:

- information about the job that is running in this address space (shown in Table 11-29)

- pipe information that is repeated for each pipe that is accessed by this address space (shown in Table 11-30)

Job Information

Table 11-29 shows the job information that is displayed.

Table 11-29 Job Information

Address Space Information	Description
ASID	address space number (in decimal) in which the participant is executing
JOBNAME	participant job name
JOBID	participant job ID (number)
STEP-NUM	current step number within the job (if this step accesses a pipe), or the number of the last step that accessed a pipe (if the current step does not access a pipe).

Pipe Information

Table 11-30 shows the pipe information that is displayed.

Table 11-30 Pipe Information (Part 1 of 3)

Pipe Information	Description
PIPE	pipe (dataset) name
ID	unique pipe identifier—the first 8 characters of the pipe time stamp
TYPE	participant type: • READER • WRITER
PIPETYPE	pipe type: • AUTOMATIC—Pipe was defined by using Job Optimizer Pipes rules. • MANUAL—Pipe was defined by using subsystem parameters in JCL. • STP2STP— Pipe was dynamically defined by Job Optimizer to implement parallel steps processing. • INTERNAL—Pipe is used for Job Optimizer Pipes internal purposes. • API—Pipe was defined using the Job Optimizer Pipes API (for future use). • BTCHPIPE—Pipe is a BatchPipes-compatible pipe.
LAST-ACT	time of last operation

Table 11-30 Pipe Information (Part 2 of 3)

Pipe Information	Description
STAT	participant status: • AFT-EOF—End-Of-File was sent/received by this participant. • UNC-CLOS—Close was performed following a previous error (unconditional close). • DEFRCLOS—Deferred close was performed. • UNC-DALC—Deallocation was performed after a previous error (unconditional deallocation). • IMM-ERR—An error in the pipe causes the participant to abend immediately. • OFI/DLY—For a writer: the writer writes only to the physical file. For a reader: the reader continues its process regardless of an error in the pipe, but abends at the end of the process. • IGN-ERR—An error in the pipe was ignored by this participant.
PHASE	current phase of the pipe participant—for internal use
PRT-FLG	participant-related error indications: • REJECT—Participant was unable to join the pipe for various reasons. • USRERROR—Participant encountered an error. • INTABDON—Pipe Handler abended this participant • DISTABND—Participant distributed its error to all other participants. When they receive the error, they should abend. • TIMEOUT —timeout expired, and the participant accepted that indication. • PIPEIOE—The pipe I/O Error status was encountered. • USRABEND—Participant abended during pipe processing.
PIPE-FLG	pipe-related error indications: • PIPERROR—Pipe is in error. • ABENDIMM—Pipe is in error. All participants should immediately abend. • ERRDIST—Error was distributed from another participant. • TIMEOUT—Timeout expired, and the participant was notified. • PEERH—Participant is in error, and the error was handled and distributed to the other participants. • PIPEIOE—Pipe I/O error occurred. • NO-RDRS—All readers closed before receiving End-Of-File. • PIPECNCL - pipe was canceled by the operator using the F MBOP,CANCEL PIPE= operator command.
PIPE-FL2	Second pipe related error indications: FORCTERM - Job Optimizer Pipes address space was terminated with FORCE option.
STEP-NUM	step number, within the job, that allocated the pipe
DDNAME	name of DD statement that allocates the pipe

Table 11-30 Pipe Information (Part 3 of 3)

Pipe Information	Description
HOME-SYS	name of the Pipe Home System (PHS) This is the system where the pipe is managed.
TIMESTMP	unique identifier of a pipe. Its value is determined when the first participant allocates the pipe.
FLAG1	various indications: • RMT-PRT—Remote participant. This participant runs in a system that is different from the PHS. • GLB-PIPE—Global pipe. • FAST-PIP—Global pipe with only one writer. • SINGL-RD—Fast pipe with only one reader.

Chapter 12 Job Optimizer Pipes Implementation Considerations

BMC Software, Inc., Confidential and Proprietary Information

Overview

Pipes provide means of transferring data between applications without using external storage, while allowing the applications to execute in parallel. Customarily, data is transferred between applications by using sequential data sets. Using this method, an application that reads the data cannot begin until the application that writes the data has finished execution. Pipes can be used whenever sequential datasets are used. Using pipes, applications can run in parallel and each data block can be read immediately after it is written.

An application that accesses a pipe is called a *pipe participant*. A pipe participant that writes to a pipe is called a *writer*; a pipe participant that reads from a pipe is called a *reader*. An application can read from more than one pipe and can write to more than one pipe at the same time.

The following types of pipes are supported:

- job-to-job pipes
  Pipes that are established between batch jobs running in parallel, using pipe rules and/or the SUBSYS JCL parameter

- step-to-step pipes
  Pipes that are established by Job Optimizer between steps running in parallel. You cannot control these pipes.

- BatchPipes-compatible pipes
  Pipes that are established between batch jobs running in parallel, using BatchPipes subsystem parameters

More than one type of pipe can be implemented for the same job. For example, the first step of the job may share a job-to-job pipe with another job. The second and third steps of that job may run in parallel by Job Optimizer and may share a step-to-step pipe.

Note: Implementation considerations in this chapter are relevant to job-to-job pipes only. For implementation considerations for BatchPipes-compatible pipes, and for migration from BatchPipes-compatible pipes to job-to-job pipes, see Appendix E, “Migrating from BatchPipes to Job Optimizer Pipes.”

Pipe Management

Pipes are managed by the Job Optimizer Pipes subsystem and the Job Optimizer Pipes address space.

Pipe management involves several mechanisms that serve to

- dynamically replace a sequential data set with a pipe (Dynamic Pipe Setting) thereby making the usage of pipes transparent to the JCL
- emulate sequential data set processing to applications, thereby making the usage of pipes transparent to the applications
- synchronize data transfer operations between applications
- ensure data integrity
- automatically create a sequential file containing the data written to the pipe, for subsequent processing or for backup purposes (Automated File Creation)

Dynamic Pipe Setting

Dynamic Pipe Setting means replacing references to sequential data sets with references to pipes, without changing the job’s JCL. This process is performed by searching for data sets in the job that do not reference a pipe and that can be replaced by a pipe. The decision whether a data set can be replaced by a pipe is made by looking for pipe rules matching the datasets defined in this job. For every dataset having a matching pipe rule, the reference to the dataset is replaced by a reference to a pipe. (For more information on rule search logic, see “Rule Management” on page 12-43.) If Automated File Creation is required, the file definitions are prepared at this stage.

Dynamic Pipe Setting is performed during job initiation by the Real Time Job Handler component of the Job Optimizer Pipes subsystem.

Note: In JES3 environment, Dynamic Pipe Setting comprises two phases. The first phase is performed as part of C/I processing by JES3. The second phase is performed during job initiation.

Pipe Life Cycle

The pipe life cycle is similar to a data set life cycle. It begins at allocation and ends at deallocation. Other stages in a pipe life cycle are Open, I/O, and Close.

Allocation—Pipe/Connection Definition

When a step starts execution, the allocation process passes all pipe allocation requests to the Pipe Handler component of the Job Optimizer Pipes subsystem. The pipe is defined when the first participant performs allocation by way of JCL at step initiation or by way of dynamic allocation during step execution.

When allocation has been performed successfully, the pipe is defined and the participant is connected to the pipe. Subsequent allocations to the same pipe name by other participants create additional connections to this pipe.

The allocation requests of the pipe participants can be synchronized. Participants wait during allocation until all required participants perform allocation to the pipe. For more information about the synchronization process and parameters, see “Synchronization Methods” on page 12-18.

If a reader allocating the pipe does not provide all DCB attributes (LRECL, RECFM, BLKSIZE), the allocation process is delayed after the pipe/participant connection is defined, until a writer opens the pipe and supplies all the DCB attributes. The reader is allowed to continue, using the DCB attributes that have been retrieved from the pipe.

Information about all the pipes that have been defined in the system (or in the entire Parallel Sysplex environment if working in Global mode) is kept in the Pipe list. The Pipe list contains a Pipe List Entry (PLE) for each pipe, with information about the pipe and all its participants. A pipe is defined by creating a PLE in the Pipe List.

Open—Pipe/Connection Initialization

The pipe initialization process starts when the first writer opens the pipe. Any participant opening the pipe after the pipe was initialized performs connection initialization.

If a reader opens a pipe before it is opened by the first writer, the reader cannot complete the Open process and cannot initialize the pipe. The reader performs a Deferred Open and continues execution without waiting for pipe initialization. When the reader performs its first I/O operation, it will wait for pipe initialization (if not already performed) and will perform its connection initialization.

The Open requests of the pipe participants can be synchronized. Participants wait during Open until a predefined number of participants open the pipe. Only then are the pipe and all connections initialized. For more information see “Synchronization Methods” on page 12-18.

The initialization process is performed in the Job Optimizer Pipes address space that runs in the same system in which the first writer runs. This system is called the Pipe Home System (PHS). The pipe initialization process includes creation of a Pipe Control Task (PCT) in the Job Optimizer Pipes address space, that is active as long as the pipe is active. The PCT allocates the required buffers in the data space or in the coupling facility and prepares the pipe control blocks that are used during I/O.

When the pipe initialization process is completed (immediately after the first writer arrives, or after all required participants have arrived), the pipe is Active, meaning it can service I/O requests. According to pipe rule definitions, participants can join the pipe after the initialization process is complete. For more information, see “Pipe Participants” on page 12-14. Only the connection initialization process is performed for each such participant.

When a participant closes and re-opens the pipe, partial connection initialization is performed because the participant remains connected to the pipe.

Note: A reader can re-open the pipe only if no I/O was performed before the pipe was closed (for example, an application that opens the pipe to retrieve the pipe attributes, closes the pipe and re-opens and reads the pipe).

Input/Output—Pipe is Active

When pipe initialization is complete, the pipe is Active. Participants who are connected to the pipe can perform I/O to or from the pipe. A writer will wait during I/O if the writer wants to write a block and the pipe is full. A reader will wait during I/O if the reader wants to read a block and the pipe is empty.

When Automated File Creation is required, each block that is written to the pipe is also written to the file.

When the writer finishes writing to the pipe, End-Of-File indication may be written to the pipe (as part of the Close or Deallocation process). When the pipe is empty and End-Of-File indication is written, any reader trying to read from the pipe will receive an End-Of-File indication. For detailed explanation about the End-Of-File process and parameters, see “End-Of-File (EOF) Process” on page 12-22.

Close—Pipe/Connection Termination

When a participant closes a pipe, a request is passed to the PCT to perform connection termination. The PCT writes a participant statistics record to the OUTLOG file and performs connection termination for this participant. When the last participant closes the pipe, the PCT writes a pipe statistics record to the OUTLOG file, performs pipe termination (frees the pipe buffers and all the associated control blocks), and terminates.

The Close requests of the pipe participants can be synchronized. Participants wait during Close until all the participants Close the pipe. Only then all pipe connections terminate and pipe termination is performed. For more information, see “Synchronization Methods” on page 12-18.

SMF records can optionally, according to the SMF record parameter specification in PIPPEMxx member, be generated when a participant closes the pipe and when the pipe is terminated. When a participant closes the pipe, the SMF record is generated by the participant. It contains general pipe details and participant I/O statistics. When the last participant closes the pipe and the pipe is terminated, the SMF record is generated by the PCT. It contains general pipe details, pipe statistics and I/O statistics for each participant

Job Optimizer Pipes performs partial connection termination for a participant in the following situations:

- A writer closes the pipe without sending an End-Of-File indication (according to pipe rule specifications).
- A reader closes the pipe without performing any I/O operation.

Partial connection termination is called Deferred Close. After Deferred Close, the participant remains connected to the pipe and may re-open the pipe and continue writing/reading. The actual close is performed at deallocation (for a writer, or for a reader that did not re-open the pipe) or for a reader, when a pipe is closed after re-open and I/O.

BMC Software, Inc., Confidential and Proprietary Information

A reader may be allowed (according to rule specifications) to close the pipe before receiving End-Of-File (read only part of the data). If there are no more active readers for the pipe, and no additional readers are allowed to join, this situation is called Early Readers Close. When Early Readers Close occurs, the writer does not continue writing to the pipe (there's no reader to read the data). If a Automated File Creation was requested, the writer continues working while writing the data to the file only; otherwise, the writer abends.

Deallocation—Pipe/Connection Deletion

Deallocation of a data set is performed during step termination processing (except for dynamic deallocation, that is performed by the application during execution). When Deallocation is performed, the participant is disconnected from the pipe. When the last participant disconnects from the pipe, the pipe is deleted, the PLE is removed from the Pipe List, and the pipe no longer exists.

The Deallocation requests of the pipe participants can be synchronized. Participants wait during Deallocation until all participants deallocate the pipe. Only then are the pipe and all connections deleted. For more information about the synchronization process, see "Synchronization Methods" on page 12-18.

If deallocation request was preceded by Deferred Close, Job Optimizer Pipes performs pipe/connection termination (close) before pipe/connection deletion, including sending End-Of-File indication (if required according to pipe rule specifications).

When the pipe has been deleted, the next reference to a pipe with the same name starts a new pipe.

Using Pipes

Pipes can be used whenever data is transferred by way of sequential data sets.

Pipes are identified by a pipe name. Pipe name must adhere to data set name rules. A pipe name must be unique (only one pipe with a given name can be active at any time). When a pipe replaces a data set, the pipe name is the name of the data set it replaces.

Pipe participants must run in parallel. They can run in the same system or in different systems in a Parallel Sysplex environment.

The writer of the pipe can optionally create a file containing the data that has been written to the pipe. This file creation is performed by Job Optimizer Pipes subsystem and is transparent to the application and optionally, to the JCL.

Pipes can be implemented by any of the following methods:

- pipe rules
- JCL or dynamic allocation, using the SUBSYS parameter to reference the Job Optimizer Pipes subsystem

A job can refer to any number of pipes in the same step, using any combination of these methods.

Pipe Usage through Pipe Rules

Pipes can be used through rules only (dynamic pipe setting is performed) or through rules with JCL definitions (rules are used for pipe parameters only). When pipe rules are used for dynamic pipe setting, the JCL that runs the job is not changed. When the pipe rule is loaded, the job runs with a pipe. When the pipe rule is not loaded or is held or removed, the job runs without a pipe.

A step in a job can be excluded from a dynamic pipe setting process. This process can be performed by adding NOPIPE DD statement to the step. For example:

```
//NOPIPE DD DUMMY
```

When this DD statement appears in a step, Job Optimizer Pipes does not look for rules for the data sets that are referenced by the step, and dynamic pipe setting is not performed.

The pipe rule identifies the required pipe (data set) and specifies the pipe parameters. Optionally, pipe participants can be specified in the pipe rule by identifying each participant as a reader or writer and setting participant-specific parameters. If participants are not specified in the pipe rule, every job accessing the data set is considered a participant and its type (reader or writer) is determined according to the PRTYPE subsystem parameter (if the pipe is used through JCL or dynamic allocation) or the JCL DISP parameter. All other pipe parameters are determined according to the pipe rule and the Pipe Rule defaults.

For more information about pipe rule usage, see “Rule Management” on page 12-43.

For using pipe rules with JCL definitions, see “Pipe Usage through JCL” on page 12-9.

Pipe Usage through JCL

Pipes can be used through JCL with or without a pipe rule. By coding the SUBSYS parameter in the JCL DD statement, a data set will be replaced by a pipe.

The only required parameters in the DD statement are DSN and SUBSYS.

Parameter SUBSYS should specify the Job Optimizer Pipes subsystem name (as defined in the BatchPlex member BPSSIDR parameter) and the subsystem parameter PRTYPE that contains the participant type. For more information, see “JCL/Dynalloc Parameters” on page 12-9. For example:

```
//SYSUT2 DD DSN=TEST.FILE,DISP=(NEW,KEEP),
//  SUBSYS=(MBOP,’PRTYPE=WTR’),
//  DCB=(LRECL=80,BLKSIZE=800,RECFM=FB)
```

In this example, the participant is identified as a writer by way of the PRTYPE subsystem parameter. If a pipe rule exists for this data set, the pipe parameters are determined according to the pipe rule and the Pipe Rule defaults. If no pipe rule is defined for this data set, all required parameters are taken from the Pipe Rule defaults. For more information, see “Rule Management” on page 12-43.

Pipe Usage through Dynamic Allocation

Adding parameter SUBSYS to the Dynamic Allocation parameters causes a data set to be replaced by a pipe. The coding rules are the same as in JCL. For more information on the coding rules, see “Pipe Usage through JCL” on page 12-9.

JCL/Dynalloc Parameters

Several JCL/dynalloc parameters are used by the pipe process.

DSNAME

The DSNAME is used as the primary selection criteria for pipe rule search. When a pipe is defined, because it was dynamically set or implemented through JCL, the DSNAME is used as the pipe name.

DISP (File Status)

For dynamically set pipes, the first decision about the participant type is based on the DISP parameter. Specifying NEW or MOD or specifying OLD with normal disposition other than DELETE identifies a writer. Specifying SHR or specifying OLD with normal disposition of DELETE identifies a reader. This decision can be overridden by defining the participant as a writer or a reader in the pipe rule.

For pipes that are defined by using JCL or dynamic allocation, this parameter has no meaning. The participant type is decided according to the PRTYPE subsystem parameter.

DCB (LRECL, RECFM, BLKSIZE)

The DCB parameters that are specified on the DD statement are used as the pipe attributes, unless overridden by DCB parameters during Open. For more information about supported values and defaults, see “Pipe Attributes” on page 9-30.

NCP

The NCP parameter that is specified on the DD statement is used, when a file is created, as the NCP value for the file. This value can be overridden by NCP value during Open. If no value is specified, the default NCP depends on the NOTESUP subsystem parameter specification. If NOTESUP=YES is specified, NCP=1 is used; otherwise, NCP=5 is used.

SUBSYS

The SUBSYS parameter is used when you are using a pipe through JCL. The parameter should specify the Job Optimizer Pipes subsystem name (as defined in the BatchPlex member BPSSDIR parameter) and should contain the PRTYPE subsystem parameter and optionally the NOTESUP subsystem parameter.

PRTYPE

The PRTYPE subsystem parameter defines the type of this participant. The values for this parameter are described in Table 12-1.

Table 12-1 PRTYPE Parameter Values

Value	Description
WTR	defines the participant as a writer
RDR	defines the participant as a reader

NOTESUP

Defines whether internal checkpoint/restart support is required. For more information on internal checkpoint/restart support, see “Internal Checkpointing Support” on page 12-32. The values for this parameter are described in Table 12-2.

Table 12-2 NOTESUP Parameter Values

Value	Description
YES	Checkpoint/restart support is required.
NO	Checkpoint/restart support is not required. This is the default.

Jobname, Step Name, Procedure Step Name, DD Name

These parameters are used as secondary selection criteria for pipe rule search whenever the pipe rule that matches the DSNAME that is specified on the DD statement contains explicit participant definitions. Otherwise, all jobs that use the data set that is defined in the pipe rule are considered participants in the pipe.

Pipe Rule Definition Considerations

Pipe rule parameters provide flexibility in pipe processing according to the applications that use the pipe. Specifying the appropriate parameters in the pipe rule enables using the pipe efficiently.

Pipe Parameters

Defining the pipe parameters in the pipe rule can be required to

- override the number of buffers used for the pipe (for more information, see “Pipe Rule Options Considerations” on page 12-12).
- override the default synchronization points (for more information, see “Synchronization Methods” on page 12-18)

- override the wait time definitions (for more information, see “Wait Time Control” on page 12-25)

Participant Parameters

Defining the participants in the pipe rule can be required to

- override the number of participants that must/can access the pipe, and their attributes (for more information, see “Pipe Participants” on page 12-14)
- restrict the use of a pipe to specific jobs, job steps, and DD names
- override the default participant type determined by the JCL DISP field

  For example, this override might be required when reading a data set with DISP=OLD because by default DISP=OLD identifies a writer.
- override the default error options definitions to allow participants to abend or continue processing when another participant fails (for more information, see “Error Handling and Distribution” on page 12-26)
- override the default EOF process options (for more information, see “End-Of-File (EOF) Process” on page 12-22)
- define whether a file should be created by the writer (for more information, see “Automated File Creation” on page 12-16)
- define specific synchronization processing for a participant (for more information, see “Synchronization Methods” on page 12-18)

Pipe Rule Options Considerations

Table 12-3 lists Pipe Rule options and references to the relevant sections in this chapter, explaining the considerations to be taken when setting values to these options. BMC Software recommends that you read “Pipe Management” on page 12-3 and “Using Pipes” on page 12-7 before implementing pipes.

Table 12-3 Pipe Rule Options (Part 1 of 3)

Rule Field	Section
RULE IDENTIFICATION	
Pipe data set name	“JCL/Dynalloc Parameters” on page 12-9
PIPE ATTRIBUTES	
Buffer number	“Pipe Tuning Considerations” on page 12-35

Table 12-3 Pipe Rule Options (Part 2 of 3)

Rule Field	Section
SYNCHRONIZATION OPTIONS	
No synchronization	“Synchronization Methods” on page 12-18
Allocation synchronization	“Synchronization Methods” on page 12-18; “Error Handling and Distribution” on page 12-26
Open synchronization	“Synchronization Methods” on page 12-18
Close synchronization	“Synchronization Methods” on page 12-18
Deallocation synchronization	“Synchronization Methods” on page 12-18; “Error Handling and Distribution” on page 12-26
MAXIMUM WAIT TIME OPTIONS [a]	
No-operation wait time	“Wait Time Control” on page 12-25
No-operation wait action	“Wait Time Control” on page 12-25
Input-Output wait time	“Wait Time Control” on page 12-25
Input-Output wait action	“Wait Time Control” on page 12-25
Synchronization wait time	“Wait Time Control” on page 12-25
Synchronization wait action	“Wait Time Control” on page 12-25
WRITER OPTIONS	
Minimum Writers[a]	“Pipe Participants” on page 12-14; “Synchronization Methods” on page 12-18
Additional Writers[a]	“Pipe Participants” on page 12-14

Create File	“Automated File Creation” on page 12-16;
	“Error Handling and Distribution” on page 12-26;
	“Restart Considerations” on page 12-32

Create File DD name	“Automated File Creation” on page 12-16
Writer Error Option	“Error Handling and Distribution” on page 12-26; “Restart Considerations” on page 12-32
Writer jobs	“Using Pipes” on page 12-7; “Pipe Participants” on page 12-14
Writer step name	“Using Pipes” on page 12-7; “Pipe Participants” on page 12-14
Writer procedure step name	“Using Pipes” on page 12-7; “Pipe Participants” on page 12-14
Writer DD name	“Using Pipes” on page 12-7; “Pipe Participants” on page 12-14
Writer signals EOF	“End-Of-File (EOF) Process” on page 12-22; “Synchronization Methods” on page 12-18
Writer error condition code	“Synchronization Methods” on page 12-18
Writer skip allocation sync	“Synchronization Methods” on page 12-18
Writer skip open lynch	“Synchronization Methods” on page 12-18

Table 12-3 Pipe Rule Options (Part 3 of 3)

Rule Field	Section
READER OPTIONS	
Minimum Readers[a]	“Pipe Participants” on page 12-14; “Synchronization Methods” on page 12-18
Additional Readers[a]	“Pipe Participants” on page 12-14
Reader Data[a]	“Pipe Participants” on page 12-14; “Synchronization Methods” on page 12-18
Reader Error Option	“Error Handling and Distribution” on page 12-26; “Restart Considerations” on page 12-32
Reader jobs	“Using Pipes” on page 12-7; “Pipe Participants” on page 12-14
Reader step name	“Using Pipes” on page 12-7; “Pipe Participants” on page 12-14
Reader procedure step name	“Using Pipes” on page 12-7; “Pipe Participants” on page 12-14
Reader DD name	“Using Pipes” on page 12-7; “Pipe Participants” on page 12-14
Reader waits for EOF	“End-Of-File (EOF) Process” on page 12-22; “Synchronization Methods” on page 12-18
Reader error condition code	“Synchronization Methods” on page 12-18
Reader skip allocation sync	“Synchronization Methods” on page 12-18
Reader skip open sync	“Synchronization Methods” on page 12-18

[a] See also “Pipe Tuning Considerations” on page 12-35.

Pipe Participants

An application that accesses a pipe is called a participant. A participant can be explicitly defined (by its job name and/or step name and/or procedure step name and/or ddname) within a pipe rule.

Each participant is identified as a writer or a reader, according to the PRTYPE subsystem parameter (if specified), the DISP JCL parameter, and the pipe rule (when it exists). If PRTYPE was specified, its value determines the participant type. Otherwise, the first decision about the participant type is made according to the DISP field. This decision can be overridden by defining the participant within the pipe rule as a writer or reader.

After a participant connects (allocates) to a pipe, it cannot change its type (for example, it cannot write to the pipe and then read from it), unless the participant disconnects from the pipe (deallocates) and reconnects (allocates) again using its new type.

Multiple Readers/Writers Support

A pipe can be accessed concurrently by a number of participants. Several writers can write data concurrently and/or several readers can read data concurrently.

The number of participants that can access the pipe is controlled by pipe rule options Minimum Writers, Additional Writers, Minimum Readers, and Additional Readers. These options specify the minimum number of participants of each type that are required to join the pipe and whether additional participants can join the pipe. The pipe does not become active until the minimum writers/readers joined (see “Synchronization Methods” on page 12-18 for exceptions). Additional participants can join later, or not at all. If additional participants are not allowed, the minimum number of participants also becomes the maximum. If the number of additional participants is unlimited, the pipe is considered an “open pipe.” Any number of participants can join an “open pipe.”

Multiple Readers

When several readers read data from the pipe, each reader can get

- all data from the pipe (A is specified in the Reader Data option)

  Each block is read by all the readers.

- the next available block (N is specified in the Reader Data option)

  Each block is read by only one reader; each reader gets a different portion of the data.

Multiple readers are usually assigned to a pipe when the pipe replaces a sequential file that is read by multiple applications. Each reader reads all data.

If a pipe is accessed by several readers, and each reader gets all the data, no additional readers can be allowed to join the pipe (Additional Readers=N should be specified) because these readers may be missing data that passed through the pipe before they joined.

Multiple Writers

When several writers are writing data to the pipe, data enters the pipe in the order of arrival (when a writer fills a block, it is written to the pipe).

It is uncommon to have multiple writers to a pipe because standard MVS processes do not support multiple writers when using sequential files. However, multiple writers can be assigned to a pipe when the pipe replaces a sequential file that is usually created by one application and extended by another (another application adds records at the end of the file). The replacement can be implemented only if the order of the records in the pipe is not important.

If a pipe is accessed by several writers, a physical file cannot be created. For more information, see “Automated File Creation” on page 12-16.

Automated File Creation

Certain applications may require that data that is passed through a pipe also be kept in a physical file for subsequent processing or backup purposes. Data that is stored in files can also be useful in recovery situations. Job Optimizer Pipes enables the data that passed through the pipe to be written to a physical file. This process is controlled by pipe rule definitions and is transparent to the application.

The file is created by the writer because the writer has the authority to create it. A new DD statement (dynamically or manually defined) is used for file creation. When the application writes to the pipe through its standard DD statement, Job Optimizer Pipes writes the data to the pipe and the physical file (using the new DD statement). The file DCB attributes are equal to the pipe DCB attributes. When the physical file is deallocated (usually at step termination), its disposition is determined according to the DISP specifications from its DD statement. The file can then be used by subsequent application that cannot work in parallel with the writer.

The pipe readers access the data from the pipe without accessing the physical file and without performing physical I/O. The reader does not affect physical file creation or disposition. During standard processing (without a pipe), it is customary for the writer to create a file that is read and deleted by the reader. However, when using pipes with automated file creation, the file is not accessed by the reader and is not deleted at the end of the process.

A file can be created when the pipe is dynamically set or when the pipe is defined in JCL or in dynamic allocation.

The file creation process is performed when all of the following conditions have been satisfied:

- There is a matching pipe rule for a data set.
- The participant is identified as a writer.
- The Create File field in the pipe rule contains a value WTR.
- Only one writer can participate in a pipe when a file is created.

Creating a File When a Pipe is Dynamically Set

File creation is automated through appropriate rule definitions (without changing the JCL). As part of the dynamic pipe setting process, Job Optimizer Pipes adds a special DD statement named JOPFILE*x* (where *x* is a unique character) containing the parameters of the DD statement that will be replaced by the pipe. This new DD statement is used for file creation. The original DD statement is replaced with a reference to Job Optimizer Pipes subsystem. The file and the pipe will have the same data set name.

To enable file creation, ensure that the following conditions exist:

- All required conditions (mentioned above) are satisfied.
- The DD statement defining the data set to be replaced by a pipe contains all parameters that are required to allocate the file. For example:

  ```
  //TRANS DD DSN=PROD.TRANS.FILE,DISP=(NEW,CATLG),
  //  DCB=(LRECL=80,BLKSIZE=8000,RECFM=FB),
  //  UNIT=3390,VOL=SER=PRD001,SPACE=(CYL,(5,5))
  ```

Creating a File When a Pipe is Defined through JCL

File creation is performed using appropriate rule and JCL definitions. Add a new DD statement to the step containing all the parameters required to create the new data set. The name of the created file can be the same as or different than the name of the data set replaced by pipe.

Note: When the pipe is defined via dynamic allocation, the same conditions should be met. The file does not have to be added to the JCL. It can be allocated using dynamic allocation before the pipe is opened.

To enable file creation, ensure that the following conditions exist:

- All required conditions (mentioned above) are satisfied.

The DD statement defining the pipe references the Job Optimizer Pipes subsystem and contains the PRTYPE=WTR subsystem parameter. For example:

```
//TRANS DD DSN=PROD.TRANS.FILE,DISP=(NEW,CATLG),
//  DCB=(LRECL=80,BLKSIZE=8000,RECFM=FB),
//  SUBSYS=(MBOP,’PRTYPE=WTR’)
```

- A new DD statement is added, containing all the parameters required to allocate the file. For example:

```
//TRANSFL DD DSN=PROD.TRANS.FILE,DISP=(NEW,CATLG),
// UNIT=3390,VOL=SER=PRD001,SPACE=(CYL,(5,5))
```

If DCB attributes specified on this DD statement are different than the pipe DCB attributes, they are overridden by the pipe DCB attributes.

- The Create File DD Name field in the pipe rule contains the name of the pre-defined DD statement to be used for the file.

Synchronization Methods

Synchronization means holding the pipe participants at certain processing points to ensure that all participants arrive to these points successfully.

When running the applications that use files sequentially, failure of an application that accesses the file does not affect the other applications and does not cause data loss. This is different when using pipes, because the applications are running in parallel, the pipe exists for the time the applications are active, and it can hold only a certain amount of data. Any failure in an application that is participating in the pipe may cause data loss.

To prevent data loss, it might be useful to synchronize the processing of the pipe participants. For example, you can synchronize the Open process of all the participants before allowing the writer to start writing data into the pipe. Otherwise, the writer might write data into the pipe (until the pipe becomes full) while there is no reader to read it (because the reader never started or failed on JCL, for example). In this case, the data written to the pipe would be lost unless Automated File Creation was requested. As another example, it may be necessary to ensure that all data was successfully read from the pipe before the writer can close the pipe and continue with its process. Otherwise, the writer may end OK while the reader failed to read part of the data (that was lost).

The following points of synchronization are available:

- None
- Allocation
- Open
- Close
- Deallocation

Synchronization points also determine the scope of the Error Distribution. (See “Error Handling and Distribution” on page 12-26.)

Synchronization points are divided into the following groups:

- Synchronization Before I/O (Allocation, Open)
- Synchronization After I/O (Close, Deallocation)

Synchronization before I/O

Synchronization points Allocation and Open ensure that all required participants (Minimum Writers and Minimum Readers that are defined in the pipe rule) arrive and are ready to process the pipe before any data is written to the pipe:

- Allocation—Most of the allocations are performed as part of step initiation process (as a result of JCL statements), so this synchronization point ensures that the participants do not start their processes before all required participants are connected to the pipe.

- Open—This type ensures that no data is written to or read from the pipe before all required participants open the pipe. Synchronization at open applies to the first open that is performed by the applications to the pipe. If the applications close and re-open the pipe, synchronization is not performed for the re-open.

There are cases when synchronization at Allocation or Open is required between multiple participants but there is a need to exclude one or more participants from the synchronization requirements. For example, when a participant holds critical resources that cannot be held for a long time, the participant can request to be excluded from the Allocation or Open synchronization process by way of rule definitions. When the participant allocates or opens the pipe, it is counted toward the number of participants that are required to fulfill the synchronization requirements, but it is not held. The appropriate pipe processing (pipe/connection definition or initialization) is performed.

Synchronization after I/O

Synchronization points Close and Deallocation ensure proper completion of all participants that are connected to the pipe. All data that was written into the pipe was also read and processed successfully.

- Close—This ensures that all participants successfully processed all pipe data. Synchronization is not performed in the case of a Deferred Close because the pipe can be re-opened.
- Deallocation—Most deallocations are performed at step termination (according to JCL DD statements), so this synchronization point ensures that the participants did not encounter any error after closing the pipe.

Synchronization Parameters

The following pipe rule fields are relevant for synchronization processing:

- Synchronization Options—Define whether synchronization should be implemented for this pipe and, if yes, at which points of processing.
- Minimum Writers and Minimum Readers—Define how many participants from each type should be synchronized.
- Participant related skip alloc/open sync—Define whether a participant should be excluded from the allocation/open synchronization.

Synchronization Considerations

The nature of the applications participating in a pipe determine the points of synchronization to use for each pipe. For most applications, BMC Software recommends synchronization at Open and Close. Several expectations are listed below:

Note: Synchronization parameters setting affects also the Error Distribution processing. Before setting value for the Synchronization parameters, see “Error Handling and Distribution” on page 12-26.

- If it is important to ensure that all steps participating in the pipe start execution before allowing any operation to be performed on the pipe, synchronization at allocation is required.
- If it is important to ensure that all the steps participating in the pipe end normally, synchronization at deallocation is required.

- If any number of readers and writers can join the pipe at any time (it is an "open pipe"), no synchronization can be requested (a value of Y should be set in the No synchronization field).

- Values specified for Minimum Readers and Minimum Writers fields act as the number of required participants only when synchronization at allocation or open is requested.

- If the writer opens and closes the pipe more than once, synchronization at Close cannot be requested.

- If a reader is allowed to terminate after opening and closing the pipe without performing any I/O, synchronization at Close must not be specified. This is because the close performed by the reader is treated as a Deferred Close that allows the reader to re-open and perform its process. If the reader terminates without re-opening and performing I/O, the deallocation process will attempt to perform the close and will fail because of the synchronization requirement.

- Consider using the participant related Writer/Reader skip alloc/open sync when a participant holds critical resources that cannot be held for a long time.

- Special care should be taken when an application is a participant in more than one pipe (e.g., an application reads from pipe P1 and writes to pipe P2). Synchronization usually causes a participant to wait until all other pipe participants arrive at the same point. When an application is a participant in more than one pipe, waiting for synchronization in one pipe may delay the processing of participants in the other pipes connected to the waiting participant.

  Another potential synchronization problem is deadlock between applications.

Example

Application A writes to pipe P1.
Application B reads pipe P1 and writes to pipe P2.
Application C reads both pipes, P1 and P2.
Pipes P1 and P2 both require synchronization at Open.

Make the following assumptions:

- Application A opens pipe P1 for output.

- Application B opens pipe P1 for input and then opens pipe P2 for output.

- Application C opens pipe P2 for input and then opens pipe P1 for input.

This situation causes a deadlock because applications A and B wait for application C to open pipe P1 while application C waits for application B to open pipe P2.

The solution is to ensure that applications perform operations on the pipes in the same order:

- — If synchronization is required at open or close, the application should open or close the pipes in the same order.
- — If synchronization is required at allocation or deallocation, the DD statements of the files must appear in the same order in all the participants' JCL.

Another solution is to let one of the participants skip the open synchronization process. The participant is still counted towards the number of participants required for synchronization but does not wait. This situation prevents the deadlock. BMC Software recommends letting the reader skip the synchronization process to prevent data loss. If the writer skips the synchronization process, it may start writing to the pipe before the reader arrives. If the reader will not join, the data will be lost.

End-Of-File (EOF) Process

When sequential data sets are used, MVS sets an End-Of-File (EOF) indication in the file when the writer finishes writing to the file. When the reader reads the file, MVS passes the EOF indication to the reader after the reader has read all the records from the file (to indicate that there are no more records to read).

A similar process can be performed by the Pipe Handler when pipes are used.

EOF Process for Writers

An EOF indication can be sent by the Pipe Handler on behalf of the writer during Close, Deallocation, or not at all, depending on pipe rule Writer signals EOF option specification. After sending an EOF indication, the writer cannot write additional data to the pipe.

Writer signals EOF is a participant-related option. When there are several writers, whether or not each of them will send an EOF indication depends on Writer Signals EOF option specification for each writer. Each writer can close and deallocate the pipe at a different time. The last writer closing/deallocating the pipe determines the status of the pipe. If the last writer sends an EOF indication at Close/Deallocation, the pipe is marked EOF; otherwise, the pipe is not marked EOF.

For example, assume that two writers are defined for a pipe: Writer E is defined with Writer signals EOF Close. Writer N is defined with Writer signals EOF No. When writer E closes the pipe, an EOF indication is sent to the pipe. When writer N closes the pipe, no EOF indication is sent to the pipe. The following conditions apply:

- If writer E is the last to close the pipe, an EOF indication is sent and is set in the pipe.
- If writer N is the last to close the pipe, an EOF indication is not sent and, therefore, an EOF is not set in the pipe. The EOF sent by writer E when it closed the pipe earlier indicates that writer E will not write any more records to the pipe, but does not change the pipe status.

A pipe that is marked EOF does not accept any new writer, and it is not possible to write data into such a pipe. Only reading is still allowed, until the pipe becomes empty.

EOF Process for Readers

A reader receives an EOF indication when the pipe is empty and is marked EOF. If the pipe is empty but not marked EOF (that is, writers are still active or the last writer closing/deallocating did not send an EOF indication), the reader waits for the next block.

The Reader waits for EOF pipe rule option indicates whether or not a reader can be allowed to close the pipe before receiving an EOF indication. This is a participant related option. If Reader waits for EOF Yes is specified and the reader closes the pipe before receiving an EOF indication, the reader will fail.

EOF Parameters and Considerations

The nature of the applications participating in a pipe determines the type of EOF process to use. For most applications, BMC Software recommends setting Writer Signals EOF Close and Reader waits for EOF Yes.

General Considerations

An example of a pipe in which an EOF indication is not required is an “open pipe”—a pipe that any number of writers and readers can join at any time. An open pipe can be used for transaction processing applications where the pipe is opened only once, by the first writer, and remains open, even if no participant is connected to the pipe, until closed by an operator command. Each writer connects to the pipe, writes its data (one or more transactions that may be one or more records), and disconnects. A reader connects (in parallel to the writer or after the writer disconnects), reads one transaction only (one or more records), and disconnects. The pipe remains active, waiting for writers to write more transactions and for readers to read the rest of the transactions (if any remain in the pipe).

Writer Signals EOF Option Considerations

Parameter Writer signals EOF should be set according to how the writer processes the pipe. Table 12-4 lists the possible values and when to use them.

Table 12-4 Writer Signals EOF Option

Value	When to Specify
C (Close)	If the writer opens the pipe only once, writes all the data, and then closes the pipe, specify this value. An EOF indication will be sent to the pipe when the writer closes the pipe.
D (Deallocation)	If the writer opens and closes the pipe multiple times, specify this value. An EOF indication will be sent to the pipe at Deallocation, that is the last action done by the writer.
N (No)	If the readers of the pipe do not require an EOF indication, this value can be specified.

For pipes with multiple writers, each writer can have a different specification. BMC Software recommends not defining Writer Signals EOF N (No) for some writers and Writer Signals EOF C or D (Close/Deallocation) for others. Different definitions can be used only when the order in which the writers will close/deallocate the pipe is known. This condition exists because an EOF indication is set or not set in the pipe according to the definition of the last writer who accessed the pipe. If different Writer Signals EOF definitions are used (some have N and some have C or D) and the order of the participants closing/deallocating the pipe is not known, the pipe EOF setting is unpredictable.

Reader Waits for EOF Option Considerations

Parameter Reader waits for EOF should be set according to how the reader processes the pipe. Table 12-5 lists the possible values and when to use them.

Table 12-5 Reader Waits for EOF Option

Value	When to Specify
Y (Yes)	Most readers need to read all the data. Therefore, these readers should not be allowed to close the pipe before receiving the EOF indication (that indicates the end of the data).
N (No)	Readers reading only part of the data can be allowed to close the pipe without receiving an EOF indication. If all readers close before receiving EOF indication, it is considered an “early readers close” situation. The writer will stop writing to the pipe. If a file is created with the pipe, the writer continues writing to the file. Otherwise the writer abends.

For pipes with multiple readers, each reader can have a different specification. For example, one reader may be allowed to read only part of the data (Reader waits for EOF N) while another reader has to read all the data (Reader Waits for EOF Y).

Wait Time Control

One of the main benefits of pipes is elimination of the delays that applications encounter when performing I/O to DASD or tape. Yet, when using pipes, applications can still encounter delays while waiting to produce or retrieve the data.

Job Optimizer Pipes monitors participant activity to identify situations in which a participant is waiting too long for an event to occur. Such a situation can occur because of

- scheduling problems (for example, the participants were not submitted concurrently)
- major differences in application’s nature (for example, the reader attempts to read the pipe immediately after it starts execution, while the writer executes for 30 minutes before writing to the pipe)
- problems in applications (for example, the writer is in a loop and is not writing anything to the pipe)

Several wait types are monitored by Job Optimizer Pipes. For each type, the pipe rule can define a different time limit and/or a different action to be taken when the time limit is exceeded. Table 12-6 lists the wait types that are be monitored.

Table 12-6 Wait Types Monitored

Wait Type	Description
No-operation	amount of time during which the participant did not access the pipe
Input-Output	amount of time that a participant is allowed to wait for I/O For a reader, this is the maximum time that a reader is allowed to wait for a block when the pipe is empty. For a writer, this is the maximum time that a writer is allowed to wait to write a block when the pipe is full.
Synchronization	amount of time that a participant is allowed to wait for the other participants at any synchronization point

Table 12-7 lists the actions that you can specify when one of the time limits (listed in Table 12-6) is exceeded.

Table 12-7 Wait Time Actions

Action	Description
A (Abend job)	Abend the participant that reached the limit.
O (ask Operator)	Ask the operator how to handle the participant. The operator can request that the participant be allowed to wait for another time period (WAIT) or that the participant abend (ABEND).

Wait Time Considerations

Wait time limits should be defined according to the nature of the applications participating in the pipe. The values should be set so that they will allow the application to work and share a pipe with minimum delays but still be warned when exceptional waits occur. For different cases and additional considerations, see “Pipe Tuning Considerations” on page 12-35.

Error Handling and Distribution

Applications usually run sequentially. The data retrieval application starts only after successful completion of the data creation application. When using pipes, data retrieval applications run in parallel with data creation applications, not “knowing” about failures that can compromise data integrity.

A pipe participant can encounter several types of errors during execution:

- an abend that the application encounters during its process (regardless of the pipe process)

- a failure that occurs as part of pipe processing

  For example, a writer tries to write after EOF, or a reader closes the pipe before receiving an EOF indication. In such situations, the Pipe Handler abends the failing participant. (See “End-Of-File (EOF) Process” on page 12-22.)

- expiration of one of the time limits that is defined for the pipe when the required action for such a situation is to abend the participant (see “Wait Time Control” on page 12-25)

- an error in which a step using the pipe completes with an error completion code (a completion code higher than the completion code defined in the pipe rule)

  This error is different from the preceding errors because the participant does not abend. The error is recognized only at step termination time.

To ensure data integrity, it is required that if one application fails while processing the pipe, the others will “know” about it and will be able to “decide” what to do. This is achieved through “Error Distribution.”

Error Distribution Processing

Error Distribution processing includes

- distributing an error that occurred in a participant to all other participants connected to the pipe

- deciding how each participant receiving the distributed error will act, according to the participant-type related Error Option specification

When a participant fails (with one of the preceding failures), the failing participant is considered In-Error and cannot access the pipe any more. The Error Distribution process marks the pipe In-Error, preventing new participants from joining the pipe. If the failure occurs as part of pipe processing, the pipe is also marked with I/O-Error, preventing other participants from accessing the pipe. In addition, the Error Distribution process sends an indication to all connected participants that an error occurred and indicates whether or not the error was severe.

Error Distribution can be implemented for a participant for the whole life of the pipe, or only for a partial process. This is called the Error Distribution scope.

By default, the Error Distribution scope begins when the participant opens the pipe and ends when it closes the pipe (or when it sends an EOF indication at the time of deallocation). Any error that occurs between open and close (or EOF) is distributed to all other participants. Errors occurring at allocation, or between close and deallocation, are outside the default Error Distribution scope and are not distributed.

This method ensures that data is not lost while the pipe is active (all the participants that are connected to the pipe can process the data).

The default Error Distribution scope is also the minimal Error Distribution scope; it can be increased but cannot be decreased. Increasing the Error Distribution scope is done by changing the synchronization points. (See “Synchronization Methods” on page 12-18 for a description of synchronization points.) The following conditions apply:

- When synchronization at allocation is requested, any error during allocation will be distributed.

- When synchronization at deallocation is requested, any error that occurs until deallocation (usually at the end of the step), including a step error completion code, is distributed.

When an error is distributed, all other participants who are connected to the pipe receive the distributed error. Each participant responds to the error according to its type (reader or writer), the Error Options that are defined in the pipe rule, and the type of the error.

Readers Error Option

The Readers error option rule parameter defines how readers will act when they receive a distributed error. Table 12-8 lists Readers Error Option specifications.

Table 12-8 Readers Error Option

Option	Description
I (Immediate)	readers abend immediately
D (Delay)	readers continue reading from the pipe When all data is read from the pipe, the readers abend.
G (iGnore)	readers continue reading from the pipe When all data is read from the pipe, the readers successfully terminate.

The Error Option specification is ignored when the pipe has an I/O-Error. In this case, the readers abend immediately (Error option immediate is forced).

Writers Error Option

The Writers error option rule parameter defines how writers will act when they receive a distributed error and when a time limit expires and the requested action (either from the rule, or by the operator replying to the time limit exceeded message) is abend. Table 12-9 lists the Writer error option specifications.

Table 12-9 Writers Error Option

Options	Descriptions
I (Immediate)	The writers abend immediately.
I (iGnore)	This option is valid only when Automated File Creation is requested. The action of the writer is determined by the pipe status and the status of active readers: • If the distributed error is received before the writer wrote any records to the pipe, the writer abends immediately (Immediate is forced) to enable restarting the whole process. • If the pipe has an I/O-Error indication, the writer stops writing to the pipe and continues writing to the physical file only. • If the error is not an I/O-Error and active readers are ready to read data from the pipe, the writer will continue writing to both the pipe and the physical file. If the error is not an I/O-Error but there are no active readers, the writer will stop writing to the pipe and will continue writing to the physical file only. • If one of the time limits expires for the writer, and the action requested (from the rule or by the operator as a reply to the time limit exceeded message) is abend, the writer stops writing to the pipe and continues writing to the file.

Error Distribution Parameters

The following pipe rule fields are relevant for Error Distribution processing:

- Allocation synchronization—increases the Error Distribution Scope to allocation
- Deallocation synchronization—increases the Error Distribution scope to deallocation
- Participant-related error condition code—defines which step condition codes should be treated as error for error distribution purposes

  This parameter is applicable only when Deallocation synchronization is defined, and the deallocation is performed as part of step termination (not dynamic deallocation).
- Participant type error option—defines how participants act when there is an error on the pipe or when receiving a distributed error

For detailed explanations, see “Readers Error Option” on page 12-28 and “Writers Error Option” on page 12-29.

Error Distribution Parameters Considerations

Error Distribution considerations are divided into Error Distribution scope parameters and Error Options parameters.

Error Distribution Scope Parameters Considerations

Error Distribution scope can be increased using the Allocation/Deallocation synchronization parameters. When setting these parameters for error distribution purposes, see “Synchronization Methods” on page 12-18 for synchronization considerations.

When increasing the Error Distribution Scope until deallocation, the step condition code can be treated as an error according to participant error condition code parameter. This process enables notifying all participants that are connected to the pipe when a participant ends with an error condition code. This situation is not applicable for pipes that are deallocated by using dynamic deallocation.

Error Option Parameters Considerations

Specification of Error Option parameters should ensure data integrity and should allow maximum parallelism and prevent massive restarts. The decision what to specify in the error option fields should consider also whether Automated File Creation is requested or not.

When Automated File Creation is requested, the file can be used as a “safety net” for data. The writer application can ignore errors in the pipe and/or readers and continue writing to the file and optionally to the pipe (if the pipe is not in error and there are readers to read it). This process maximizes the amount of concurrent processing and enables use of the physical file to restart only the abending readers. Error Option specifications for readers and writers should be based on the following considerations:

- If the file can be used as a “safety net,” the Writer error option should be set to G (ignore) to enable the writer to continue writing to the file, and optionally to the pipe, even if an error occurs in the pipe process or in one or more readers. If multiple readers use the pipe, each reading all the data (Reader data A (all)), the Reader Error option should be set to G (ignore) to enable the other readers to continue using the pipe. When all the participants terminate, the abending reader can be restarted using the physical file. If the pipe becomes unusable, Job Optimizer Pipes abends all the readers regardless of their Error option specification and the writer can continue its process, writing only to the physical file. When the writer terminates, all the readers can be restarted using the physical file.

- If Automated File Creation is requested but the readers must use the pipe and not the file, both the Reader and Writer error option parameters should be set to I (immediate) to restart all the participants and repeat the process.

Note: If the writer of a pipe with a physical file abends and its Writer error option is G (ignore), Job Optimizer Pipes overrides the Error option specification for readers and forces all readers to abend immediately. This action is done because the physical file (the “safety net”) may be incomplete and the whole process may need to be repeated.

When Automated File Creation is not requested, the writer application cannot ignore errors in the pipe (Writer error option I (immediate) is forced). Specify the Reader error option based on the following considerations:

- Any abend in one of the participants can cause data loss (a buffer was not written to the pipe or a buffer already read was not fully processed). If this situation is not acceptable, the Error option for both readers and writers should be set to I (immediate) and, in case of error, the whole process should be repeated.

- BMC Software recommends that you *not* set the Reader error option to G (ignore) because there will be no indication in the reader’s job that there was a problem in the pipe and that the job may have to be restarted. BMC Software recommends Error option D (delay) in order to allow the reader to read all of the data from the pipe but still get notification that there was a problem in the process.

Internal Checkpointing Support

Job Optimizer Pipes supports internal checkpointing by the participants, using MVS NOTE and POINT services. This support is available only when Automated File Creation was requested and subsystem parameter NOTESUP=YES was specified in the pipe DD statement. The checkpoint is the position of the block in the file (exactly as when working without a pipe).

NOTE (Checkpoint) Support

NOTE returns the position of the block in the file (exactly as when writing/reading directly to/from the file). As each block is written to the file, its position in the file is kept in the pipe along with the data block. When a participant issues the NOTE request, the file block position is retrieved from the pipe and returned to the requester. The NOTE request process does not require any access to the file.

POINT (Restart) Support

POINT requests to re-position on a specific data block (exactly as when writing/reading directly to/from the file). This request is a non-sequential action while the pipe is processed sequentially. Therefore, POINT cannot be performed against the pipe (for example, the POINT may request to re-position on a block that was already deleted). When POINT is performed by a pipe participant, the following actions are performed by the Pipe Handler:

For a writer–the writer stops writing to the pipe and continues writing only to the file. All readers connected to the pipe abend.

For a reader–The reader abends. The decision for all other participants is based on the Error Option specified in the pipe rule.

Restart Considerations

When there is an error in one of the participants, one or all the participants should be restarted, with or without pipes. The decision how to perform the restart depends on:

- What abended (a reader or writer).
- Whether Automated File Creation was requested.
- The Error Option specified for the readers/writers.

When a writer abends

- The data in the pipe is incomplete. Both readers and writers should be restarted. The restart can be with or without a pipe.

When a reader abends:

- If Automated File Creation is not requested, the writers will abend, causing the data in the pipe to be incomplete. Both readers and writers should be restarted. The restart can be with or without a pipe.

- If Automated File Creation is requested and the writer's and reader's Error Option is G (ignore), the writer will finish writing to the physical file and optionally to the pipe (if there are active readers). The abending readers can be restarted later using the physical file created by the writer.

- If a physical file is created and the writer's error option is I (immediate), the writer will abend. Restart is required for all the participants. The restart can be with or without a pipe.

Restarting the Participants

Restarting the participants without a pipe requires holding the pipe rule to prevent dynamic pipe setting. For command syntax and explanations, see "Holding or Releasing a Loaded Rule" on page 11-13. For pipes defined via JCL, restarting the participants without a pipe requires also changing the JCL to remove the reference to the Job Optimizer Pipes subsystem and delete the new DD statement that was added for Automated File Creation (if it exists).

Multi-System and Parallel Sysplex Support

Job Optimizer Pipes services can be provided in a single system or multi-system (Parallel Sysplex or non-Parallel Sysplex) environment. The Job Optimizer Pipes address space must be activated in each system in which jobs or steps using pipes will be run.

When working in a single system or in a multi-system non-Parallel Sysplex environment, Job Optimizer Pipes works in Local mode. In this mode, each system is managed separately. Applications running in parallel and sharing a pipe cannot run in different systems.

In a Parallel Sysplex environment, Job Optimizer Pipes can work in Local or Global mode. In Local mode, Job Optimizer Pipes manages each system separately. In Global mode, all the systems are handled as one system, allowing applications running in parallel and sharing a pipe to run in different systems. The Coupling Facility (CF) is used by all systems to share data (for example, the Pipe List).

When all the participants of a pipe run in the same system, the pipe is called a Local pipe. (Local Pipes can be used in either Local or Global mode.) When the participants of a pipe run in different systems, the pipe is called a Global pipe. (Global pipes can be used only in Global mode.)

A Global pipe is managed by Job Optimizer Pipes in the system where the first writer of the pipe is executing. This system is called the Pipe Home System (PHS). The Pipe Control Tasks (PCT) operates in the Job Optimizer Pipes address space of that system. Participants of a Global pipe that run in the PHS are called Local participants. Participants of a Global pipe that do not run in the PHS are called Remote participants. All participants use Parallel Sysplex services to access the pipe buffers.

Note: Unless otherwise stated, all descriptions in this section concerning the Parallel Sysplex environment assume that work is performed in Global mode.

Parallel-Sysplex Implementation Considerations

When using Job Optimizer Pipes in parallel Sysplex environment, the following considerations apply:

- Job Optimizer Pipes uses the Coupling Facility (CF) to share data (for example, the Pipe list, data blocks of Global pipes). Because the Coupling Facility is used by other MVS components (for example, JES2, VTAM and DB2), using Global pipes can affect the performance of those components. Coupling Facility response time has a major impact on the performance of Global pipes.

- Because all systems in a Parallel Sysplex environment are treated as one system by Job Optimizer Pipes, all Job Optimizer Pipes address spaces in a Parallel Sysplex environment must have identical configurations. Each Job Optimizer Pipes address space should use exactly the same set of rules, the same Pipe Policy list, the same installation parameters, and so on. The Job Optimizer Pipes address spaces must be initialized with modules that have the same level (and share the same Load library, if possible). Operator commands (for example, Rule Load, Reload PGM, Set Mode) that can affect Job Optimizer Pipes configuration or operation mode are distributed by the Job Optimizer Pipes address space receiving the command to all other Job Optimizer Pipes address spaces in the Parallel Sysplex environment. For more information see Chapter 11, “Command Scope” on page 11-3.

- JES3 considerations:

  In a JES3 complex, there is one Global processor and up to 31 Local processors. When a job is submitted to JES3, the first process it goes through is called Converter/Interpreter (C/I). As part of the C/I process JES3 analyzes JCL statements and converts them to internal control blocks. This process can occur within the JES3 address space or in a dedicated address space called FSS, and it can be performed in any system in the complex. After the job passed the C/I process, it can be executed on any system in the complex. (A job that is submitted on system A can go through the C/I process on system B, and be executed on system C.)

  The dynamic pipe setting is performed in two phases. The first phase is part of the C/I process (using JES3 exit IATUX04) and the second phase occurs when the job starts executing in a dedicated initiator. Each phase requires that the same set of rules will be active, to allow consistent operation.

  As those two phases may occur on different systems, Job Optimizer Pipes must be activated in each system where jobs using pipes will be converted or run, and the same set of rules should be loaded by all Job Optimizer Pipes address spaces.

Pipe Tuning Considerations

Repeated occurrences of “wait time exceeded” for a pipe, or delays in pipe processing, can indicate the need to tune the pipe and/or its participants. The following situations can be identified:

- When participants wait at various synchronization points, the following potential problem areas should be examined or considered:

- scheduling problems

- job JCL, to verify that the required steps are executing in parallel

  For example, if Step 1 of Job A and Step 5 of Job B are participants of the same pipe, these two steps should run in parallel. If the jobs are submitted concurrently, Step 1 of Job A will be delayed until Step 5 of Job B begins executing.

- application logic, to verify that the nature of the applications is similar

  For example, make sure that all participants open the pipe at about the same stage of the process.

- deadlocks as described in “Synchronization Methods” on page 12-18

- When a writer is waiting too long for I/O, it may be necessary to increase the number of pipe buffers or to add an additional reader (if Reader data N (next) is or can be used). If Reader data N cannot be used, consider increasing the I/O wait time limit.

- When a reader is waiting too long for I/O, the following points should be considered:

  — When Reader data A (All) is specified, faster readers may wait for I/O because slower readers have not finished reading data from the pipe. If the speed of the readers cannot be balanced, you can create a physical file along with the pipe, remove from the pipe those readers who cause the delay, and let them run later by using the physical file.

  — When Reader data A (All) is used, all readers may wait for I/O because the writers are too slow. You can add writers to increase the data transfer rate.

  — When Reader data N (next) is used, some or all of the readers may wait for I/O because the writers are too slow or because there are too many readers. You can reduce the number of readers or add writers to increase the data transfer rate.

  — If the preceding actions cannot be performed, consider increasing the I/O wait time limit.

- If the No-Operation wait time is exceeded, examine the application logic to determine whether large time intervals occur between accesses to the pipe. If this is the case, increase the No-Operation wait time limit.

- Parallel Sysplex considerations:

  — When Global pipes are used in a Parallel Sysplex environment, the participants run on different systems and use Parallel Sysplex resources, mainly the Coupling Facility (CF). A Global pipe is always slower than a Local pipe (where all the participants run on the same system and access pipe buffers directly).

  — When pipe participants (Local or Remote) are running on systems with different CPU capacity, real storage capacity, or CPU utilization, pipe access speed will not be balanced. Participants running on the stronger or less-used systems may have to wait for the other participants.

System Tuning Considerations

When parallel processing is used, resource usage in the system changes. The main changes are:

- I/O utilization is reduced (when pipes are used).
- CPU utilization at specific points is increased, mainly because of parallel processing.
- The usage of other resources (for example, work areas, initiators, tape drives) is changed because of parallel processing.

The following issues should be considered when using parallel processing:

- Parallel processing changes resource usage in the system. All resources that are required for all the jobs running in parallel must be available when the jobs are started.

  More resources (for example, initiators, tape drives, CPU) are required for shorter time periods. These resources become available after the jobs running in parallel finish execution. When running jobs in parallel using job-to-job pipes, resource usage in the system in which the jobs will run should be reviewed to ensure that all required system resources will be available when the jobs are scheduled to run.
- It may be necessary to reschedule jobs to accommodate changes in resource utilization brought about by parallel processing.
- In a Parallel Sysplex environment, all systems in the Sysplex where pipes will be used should be tuned. Pipe performance can be affected by differences in system utilization of systems that are running pipe participants.

- In a non-Parallel Sysplex environment, all parallel jobs must run in the same system. This situation might cause differences in workloads between the systems and might cause work to be redistributed between the systems.

Example

Each of two job streams contains three jobs. In both job streams, the first job creates a file that is read by the second job, that creates a file read by the third job. Both job streams run in parallel. Assuming that each job works for an hour, the processing of all jobs (with the two streams in parallel) will last about three hours.

When pipes are used to replace the files in each stream, the three jobs of each stream will run in parallel for about one hour. Because both streams run in parallel, all six jobs of the two streams will run in parallel during the same hour.

However, there may not be enough system resources available for all six jobs to run in parallel. For example:

- There may not be six available initiators.

- There may be enough workspace to run the three jobs in each job stream simultaneously but not enough workspace to run all six jobs simultaneously.

- CPU utilization may be so high when running all six jobs simultaneously that processing is significantly slowed down.

In a Parallel Sysplex environment, the jobs in a pipe collection can be scheduled in different systems in the Sysplex. The job scheduling definitions of the job in these two job streams can be changed to allow them to execute in different systems in the sysplex to enable the parallel processing.

In a non-Sysplex environment, all the jobs in a pipe collection should be scheduled in the same system. To allow pipe usage, the job scheduling definitions of the second job stream should be changed so that the second job stream starts only after the first job stream finishes execution. In this way, each job stream will have the necessary resources available.

In this example, each job stream will work for about one hour. Processing the two streams will require about two hours. This still saves an hour from the processing time that is required if pipes were not used. The saved hour is available for other processing.

Generated Abends

Job Optimizer Pipes subsystem abends a participant in any of the following situations:

- it encounters an error.
- a time limit has been exceeded, and abend is requested (see “Wait Time Control” on page 12-25).
- it receives a distributed error, and the receiving participant should abend.

The abend is preceded with error messages explaining the abend reason. Depending on the processing stage, the application terminates with one of the errors that are listed in Table 12-10.

Table 12-10 Generated Abends

Processing Stage	Abend Description
Conversion	The application terminates with a JCL error.
Allocation	If using a pipe rule or JCL, the application terminates with a JCL error. If using dynamic allocation, the application receives an error return code.
Open	The application terminates with abend S013-C0 (a subsystem file could not be opened).
I/O	The application terminates with abend S001 (an I/O error).
Close	The application terminates with abend S614-10 (a Close for a subsystem file failed). If the Close is done as part of the End-of-Task process, abend SC03 will follow abend S614-10.
Deallocation	The application terminates with abend U2420 (an internal pipe abend code).

Identifying Job-to-Job Pipes Candidates

Job-to-job pipes candidates are identified according to application and data set aspects. When identifying the best candidates for job-to-job pipes, consider the costs, benefits, and overall effects of parallel processing on the production environment.

Application Considerations

The following items should be considered when choosing the applications:

- The applications are part of a predecessor/successor relationship in which the predecessor application writes data to a data set that is then read by the successor application. Both the writer and reader applications process one/several record(s) at a time, sequentially, until completion. This should be the only dependency between the applications.

- Good candidates include jobs in a stream in which the first job creates a data set that is then read by all the successor jobs. The first job tends to create a bottleneck since the other jobs must wait for it to finish. Replacing such data sets with pipes enables the successor jobs to run in parallel with the first job and significantly reduces overall processing time.

- Applications that use pipes should only use standard I/O processing. The writer application should only write data to the file, but not update or read the file, and a reader should only retrieve the data (but not update it).

- Additional benefits can come from an application that reads a record from a sequential data set, processes it and writes it to another sequential data set. Such an application can be connected to a pipe on both the input and output sides, thus increasing the level of parallelism. A good example of such a process is an application that merges several files and writes the output to a combined file that is then used by another application. Both the input and output of such an application can move through pipes.

  An opposite example is a SORT utility that can only write output after reading and processing all the input. This type of application can be connected to a pipe for either input or output (but not both), to save I/O time.

- In a Parallel Sysplex environment, applications running in different systems in the Sysplex can be considered for sharing a pipe. In a non-Parallel Sysplex environment, applications can use pipes only if they are running in the same system. If one of the applications must, for some reason, run in another system, pipes cannot be used.

Data Set Considerations

The following items should be considered when choosing data sets that are candidates for being replaced by pipes:

- Only physical sequential data sets (DSORG=PS) can be replaced by pipes. The record format (DCB parameter RECFM) can specify any valid format except Undefined or Spanned.

- Using pipes to replace large data sets (possibly placed on tape) that are created (written) once and then accessed (read) several times can be especially beneficial. Pipes can eliminate the large number of I/O operations (and tape mounts) that would otherwise be required to access the data set by each application.

- Data sets that exist for the life of several jobs and are then deleted are natural candidates for pipes because the data is only needed temporarily.

- The requirement that a permanent data set be created by an application does not preclude the use of a pipe for this data set. Although data in pipes is held temporarily, the same data can be automatically written to a physical file.

Resource Usage Considerations

To enable parallel processing and achieve efficient utilization of all resources, the following items should be considered:

- Time periods when the CPU is under-utilized are good candidates for parallel processing. These periods may be the result of production delays caused by jobs that are candidates for pipes.

- When running applications in parallel instead of sequentially, additional resources are required, but for a shorter period. Therefore, when planning for parallel processing, the additional resource usage (more CPU, more initiators, additional tape drives, more work space, etc.) should be evaluated to prevent delays.

- When running in a non-Parallel Sysplex environment, all the pipe participants must run in the same system. This might cause a shift in workload between systems.

See also “System Tuning Considerations” on page 12-38 for resource utilization considerations.

Rule Management

Most Job Optimizer Pipes decisions are driven by parameters specified in user-defined rules (called Pipe Policy Rules). These rules reside in Pipe Policy tables that are members in a library (partitioned data set). To be used by Job Optimizer Pipes, rules must be loaded into memory. After they are loaded in memory, they can be replaced, held or deleted.

When working in Local mode, rule management is performed for each system separately. When working in Global mode, rule management can be performed automatically in all the systems where Job Optimizer Pipes is active.

Rules are used for dynamic pipe setting and for pipe management.

The following paragraphs contain information about the use and management of rules.

Automatic Loading of Rules

When the Job Optimizer Pipes address space is started, it reads the Pipe Policy list from the member referenced by BatchPlex member PIPELIST parameter. The Pipe Policy List contains a list of the Pipe Policy tables to be automatically loaded to memory by Job Optimizer Pipes. Each line in the list has the following format:

```
RULEMEM=table,RULELIB=library
```

Table 12-11 lists the parameters and descriptions for the Pipe Policy List Line.

Table 12-11 Pipe Policy List Line Parameters

Parameter	Description
table	Pipe Policy Table (member) name
library	library where the Pipe Policy table resides

For every line in the list, Job Optimizer Pipes loads rules from the specified Pipe Policy table.

Manual Loading/Reloading of Rules

Additional Pipe Policy tables can be loaded to memory, and Pipe Policy tables can be replaced in memory while the Job Optimizer Pipes address space is active. This process can be accomplished by using the MAINVIEW Batch Optimizer Dialog or by using operator commands.

In Global mode, a request to load a Pipe Policy Table by way of the MAINVIEW Batch Optimizer Dialog is distributed to all systems where Job Optimizer Pipes is active. When rules/tables are managed through operator commands, the commands can specify whether or not they should be distributed to other systems (the default is to distribute the command to all the other systems).

When a Pipe Policy table is loaded or reloaded, a new table is created in memory. Rule searches that began before the completion of the load process continue using the old table. When these rule searches are completed, the old table is deleted from memory. Rule searches that begin after the completion of the load process use the new table.

Rule Order

Pipe rules can be defined in several Pipe Policy tables. The order of the rules in the Pipe Policy Table and the order of the lines in the Pipe Policy List is not important.

When rules are loaded into memory, they are sorted according to Best Match Order and an indexed list is built. When multiple Pipe Policy tables are loaded (as defined in the Pipe Policy list), the Best Match Order sort applies to all the rules from all the tables. When a Pipe Policy Table is added or replaced, the rules are merged with the rules in memory according to Best Match Order.

Best Match Order

Best Match Order sorts the rules according to Data set/Pipe name (alphabetic values) in descending alphabetic order. Table 12-12 lists the Data Set Name Sort Order.

Table 12-12 Data Set Name Sort Order

Parameter	Order	Description
DATASET	9–0 *Z–A* ? *	sort order is reverse (descending) alphabetic order (i.e., ABC* will appear before AB*)

Rules with specific criteria are placed before more generic rules. This way, Job Optimizer Pipes can locate the rule that meets exact search criteria quickly. If no “best match” rule is located, generic rules at the end of the list can be used (similar to defaults).

Rules with the same sort parameter values are placed in the list according to their load time sequence.

Rule Search Algorithm

A rule search is performed on the rules that are loaded into memory.

Pipe rules are used for the following purposes:

- Dynamic Pipe Setting—determine whether job-to-job pipes should be used for data sets accessed by a job and, if so, what pipe parameters are used.

- Find the pipe parameters for job-to-job pipes that are defined through JCL (by way of the SUBSYS=JCL parameters) or through dynamic allocation.

- BatchPipes phased migration-determine whether a BatchPipes pipe should be processed as a Job Optimizer Pipes BatchPipes-compatible pipe. For more information about BatchPipes phased migration, see “BatchPipes to Job Optimizer Pipes Migration” on page E-2.

The search process is always performed for a specific data set and is activated for the job that accesses that data set. The first target is to find a pipe rule with a matching data set name (it can be a generic pipe rule).

- If the matching pipe rule contains specific participant definitions (for example, jobnames, ddnames), they are checked against the active job. If a match is found, the rule is used.

- If the matching pipe rule does not contain specific participant definitions, only the data set name determines whether or not the rule is used.

To determine whether a data set should be replaced with a job-to-job pipe, a matching pipe rule must be found. When a matching rule has been identified, the data set is replaced with a pipe. All the rule's defined parameters are used to decide the pipe characteristics. Values for parameters that are not defined within the rule are taken from the Pipe Rule Defaults (see "The Pipe Rule Defaults contain default values for all rule options. These defaults are used to supply missing values when a pipe rule is used, or to supply all the pipe parameters when a pipe is used by way of JCL and there is no pipe rule for this pipe." on page 12-46).

When a pipe is defined in JCL and a pipe rule is found, all of the rule's defined parameters are used to decide the various pipe characteristics. Values for parameters that are not defined within the rule are taken from the Pipe Rule Defaults (see "The Pipe Rule Defaults contain default values for all rule options. These defaults are used to supply missing values when a pipe rule is used, or to supply all the pipe parameters when a pipe is used by way of JCL and there is no pipe rule for this pipe." on page 12-46). If a rule is not found, only the Pipe Rule Defaults are used to decide the pipe characteristics.

Pipe Rule Defaults (JOPDEFRL)

The Pipe Rule Defaults contain default values for all rule options. These defaults are used to supply missing values when a pipe rule is used, or to supply all the pipe parameters when a pipe is used by way of JCL and there is no pipe rule for this pipe.

The Pipe Rule Defaults reside in pipe rule JOPDEFRL in Pipe Policy Table JOPDEFRL. This is a special pipe rule.

JOPDEFRL must be loaded during the Job Optimizer Pipes address space initialization process. The default Pipe Policy list that is delivered as part of the installation contains a line for default rule JOPDEFRL. This line is not shown when editing the Pipe Policy List by using the MAINVIEW Batch Optimizer Dialog. Removing this line from the Pipe Policy List will result in Job Optimizer Pipes initialization failure.

JOPDEFRL is excluded from the rule search process. It is used only to fill in missing values when a pipe rule has already been found or for setting the pipe and participant parameters when a pipe is defined by using JCL or dynamic allocation and no matching rule is used.

JOPDEFRL is delivered as part of the installation package. After installation, this rule should be adjusted according to site requirements. All parameters in this rule must be specified.

BMC Software, Inc., Confidential and Proprietary Information

BMC Software recommends that parameters for which the site has a standard (for example, to synchronize at Open/Close) be defined in the Pipe Rule Defaults only. If a specific data set requires different values (for example, synchronize at Alloc/Dealloc), only the rule for this data set should contain specifications for the non-standard values of these parameters. This method helps manage the site’s standards by concentrating them in one rule, making it easy to change them as required.

The Pipe Rule Defaults can be modified and activated (replaced) during Job Optimizer Pipes processing. (For detailed explanations see Chapter 9, “Job Optimizer Pipes Dialog.”) Changes made to the Pipe Rule Defaults affect all rule searches that are performed immediately after the Pipe Rule Defaults are activated (replaced).

BMC Software, Inc., Confidential and Proprietary Information

Chapter 13 Job Optimizer for DB2

This chapter explains what Job Optimizer for DB2 is, what it does, and how it works. It also explains system requirements. This chapter discusses the following topics:

BMC Software, Inc., Confidential and Proprietary Information

BMC Software, Inc., Confidential and Proprietary Information

Overview

Job Optimizer for DB2 provides support for batch workloads that involve DB2 processing. Job Optimizer for DB2 intercepts and processes DB2 steps that reference DB2 tables, and determines which steps would benefit from parallelism. Job Optimizer for DB2 also provides workload management, optimization, and sysplex enablement.

Support for Batch Workloads that Involve DB2

Job Optimizer for DB2 is invoked when it encounters a job step that involves DB2 processing. Job Optimizer for DB2 is a component of Job Optimizer and provides DB2 step constraint information.

Parallelism for Job Steps that Reference DB2

Job Optimizer for DB2 analyzes DB2 job steps and identifies the DB2 dependencies of those steps. It provides this information to Job Optimizer. Job Optimizer determines which job steps can be run in parallel on a single MVS image, or can be distributed among multiple MVS images for parallel processing. For example, if two steps of the job update the same DB2 table, Job Optimizer prevents these two steps from running concurrently, or in parallel with each other, but not separately from other steps. However, if the DB2 steps update/read two non-related DB2 tables or read the same and have no other constraints, Job Optimizer may schedule those steps to run in parallel.

Parallelism or concurrent processing is not allowed for DB2 steps that contain

- remote data access
- DB2 plans bound to packages with LOCATION=*
- DB2 plans bound to packages with COLLECTION ID=* and NAME=*
- IBM DB2 utilities
- dynamic SQL
- non-compatible table accesses
- RRSAF plan names that begin with the character “?”

Workload Management and Optimization for DB2 Jobs

Job Optimizer performs workload management and optimization for DB2 jobs by distributing job steps among multiple MVS images for parallel processing. Before this can occur, the DB2s on the MVS image must be part of an active DB2 data sharing group. Job Optimizer for DB2 analyzes the job steps that reference DB2 and identifies job steps that can be targeted to run in parallel on other MVS images whose processing capabilities are best suited to running those job steps. Job Optimizer takes advantage of idle resources for faster throughput of job steps. For example, parallel processing of steps that reference DB2 could occur on a single MVS image for non-data sharing DB2 configurations and could occur on multiple MVS images with DB2 data-sharing configurations.

How Job Optimizer for DB2 Works

Job Optimizer for DB2 works by providing Job Optimizer with a list of DB2 data constraints and access control information. DB2 job steps are intercepted and scheduling information is passed to MAINVIEW Batch Optimizer. MAINVIEW Batch Optimizer provides support for DB2 job steps through the following features:

- data capture and history recording capability
- analysis and qualification feature
- constraint analysis
- policy specification facility

Data Capture and History Recording Capability

Job Optimizer for DB2 identifies and captures the following information about a DB2 batch job from the DB2 plan(s) opened by each job step:

- DB2 ssid
- name of package and version, if using DB2 packages
- name of the DB2 plans

Job Optimizer for DB2 also records information about the specified processing policies of the DB2 batch jobs. Job Optimizer for DB2 saves this information in the history data set for data capture and history recording.

Job Optimizer builds a profile of a batch job, based on statistics and the history capture information gathered over time, for these purposes:

- to predict how the job steps behave
- to determine which job steps are good candidates for parallel processing, based on the profile

Analysis and Qualification Feature

The Job Optimizer for DB2 analysis and qualification feature determines any changes to the execution of DB2 batch workloads that would provide optimization. Job Optimizer for DB2 identifies the candidate jobs selected for recommended DB2 batch workload execution changes, based on statistics and information about processing options gathered over time. The recommended DB2 batch workload execution changes may include the following tasks:

- redirecting jobs to other processors in the sysplex
- converting batch jobs to a pipeplex for parallelism
- facilitating batch workload balancing across multiple systems
- allowing two or more job steps to run in parallel without application or allowing JCL changes

Constraint Analysis

Job Optimizer for DB2 provides constraint information to Job Optimizer to control the serialization and parallelization of DB2 job steps. The data capture contains plan and package information that is used to derive the DB2 step constraints. During the building of constraint information, Job Optimizer serializes DB2 steps if the following conditions exist within a DB2 step:

- Dynamic SQL exists in the job step.

- DB2 plan information is not available.

- There are constraints from the Job Optimizer component of MAINVIEW Batch Optimizer (for example, manual overrides).

- The job step contains remote data access.

- The job step involves IBM DB2 utility software.

- The job step in which the PLAN has PKLIST(S) bound to it with LOCATION=* or remote LOCATION name, or PKLIST(S) with COLLECTION ID=* and NAME=*.

- Certain SQL statements are used in the job step.

For more information about serialization of job steps and data constraints between job steps, see Appendix D, “Job Optimizer for DB2 and SQL Statements.”

DB2 Policy Processing

Job Optimizer for DB2 policy information can be defined at the global and the DB2 group levels. You can also define what action Job Optimizer should perform for the selected batch job.

Understanding the DB2 Policy Member in the Control Data Set

The installation and customization of Job Optimizer creates a control data set which stores the MAINVIEW Batch Optimizer’s definitions, as follows:

- BatchPlex definitions
- job policy definitions
- data policy definitions
- User Control Facility (UCF) definitions

During the installation and customization of Job Optimizer for DB2, the DB2 policy definitions are saved as members in the control data set. This DB2 policy may be activated by issuing the DB2 Policy Activate command; see “Activating Job Optimizer for DB2 Processing” on page 13-27.

Policy Specification Facility

A DB2 policy may be created, edited, and viewed, by using the MAINVIEW Batch Optimizer dialog; see “Creating Policies for Job Optimizer for DB2” on page 13-16 for more information. A DB2 policy specifies the criteria that must be met before Job Optimizer for DB2 provides performance processing for a batch job. You can also define what action Job Optimizer for DB2 should perform for the selected batch job.

In addition, you can override the default Job Optimizer for DB2 PLAN NAME and AUTHID under which Job Optimizer for DB2 executes.

BMC Software, Inc., Confidential and Proprietary Information

DB2 Policy Information

A DB2 policy contains global option statements and DB2 group option statements. GLOBAL statements are applied to all DB2 groups that do not have DB2GROUP values specified. Only the GROUP_PLAN, GROUP_AUTHID, and VERSIONS used by the DB2 plan for Job Optimizer for DB2 can be specified globally. If no global or DB2 GROUP_AUTHID is specified, the default install SYSADM ID will be used. If global or DB2 GROUP_AUTHID is specified, ID must have SYSADM authority. The DB2 group option statements let you specify actions to be taken against a specified user plan or table. The DB2 group options also let you override the default values or specified global values.

Global Option Statements

The main purpose of the group options is to let you specify global options for Job Optimizer for DB2 authorization ID and plan.

The valid global option statements are:

- GROUP_AUTHID=
- GROUP_PLAN=
- VERSIONS=
- TIME_LIMIT=

GROUP_AUTHID and GROUP_PLAN are the authorization ID and the plan name that is used to bind the Job Optimizer for DB2 plan. The plan is used to examine the DB2 catalog and must be bound on each DB2 that will participate in Job Optimizer for DB2.

VERSIONS is the number of versions of a plan or package that Job Optimizer will examine.

TIME_LIMIT specifies the amount of elapsed time to spend in gathering DB2 constraint information. The default is 5 minutes. The values can range from 1–60 minutes. If the time limit is exceeded during DB2 analysis, a DB2 constraint is applied.

The default value, if not specified, for the GROUP_AUTHID= statement is install SYSADM. The default value, if not specified, for the GROUP_PLAN= statement is MBOPLAN. The default number of versions is 5.

If you specify a different value for the GROUP_AUTHID global option, it overrides the default value of install SYSADM for that DB2. The GROUP_AUTHID must have SYSADM authority. If you specify a different value for the GROUP_PLAN global option, it overrides the default value of MBOPLAN.

Note: BMC Software recommends that the same plan name be used on all DB2s.

DB2 Group Options

The main purpose of DB2 group options is it lets you identify the DB2 subsystems or DB2 data sharing groups that Job Optimizer for DB2 is to optimize and/or provide exceptions to normal Job Optimizer for DB2 processing.

The valid DB2 group option statements are:

- DB2GROUP=
- GROUP_AUTHID=
- GROUP_PLAN=
- PLAN=
- TABLE=
- ACTION=
- VERSIONS=
- SSID=
- EXCLUDE=
- TIME_LIMIT=
- CHECK_INTERVAL=

DB2GROUP=

Specify the DB2 Data Sharing Group name in the DB2GROUP= option statement. The DB2 group option statements that follow this DB2GROUP= statement apply to that DB2 group until the next DB2GROUP= statement or the end of the DB2 policy member, whichever occurs first. For non-data sharing DB2 subsystems without a group name defined specify the DB2 SSID as the DB2 group.

GROUP_AUTHID=
GROUP_PLAN=

If the global GROUP_AUTHID is blank and no GROUP_AUTHID is specified for the DB2 group, the default value for the GROUP_AUTHID= statement is INSTALL SYSADM. If the global GROUP_PLAN is blank and no GROUP_PLAN is specified for the DB2 group then the default value for the GROUP_PLAN= statement is MBOPLAN. If the global VERSIONS is blank and no VERSIONS is specified for DB2 group, the default value is 5.

If you specify a different value for the GROUP_AUTHID option in the DB2 group option statement, the statement overrides the default value of INSTALL SYSADM that is used by the Job Optimizer for DB2 plan for that DB2 subsystem and any value that is specified for the GROUP_AUTHID global option. The GROUP_AUTHID value for the DB2 group option statement applies only to the DB2GROUP that is specified in the DB2GROUP= statement preceding it in the DB2 policy.

If you specify a different value for the GROUP_PLAN option in the DB2 group option statement, it overrides the default value of MBOPLAN and any value specified for the GROUP_PLAN global option. However, the GROUP_PLAN value in the DB2 group option statement only applies to the DB2GROUP specified in the DB2GROUP= statement preceding it in the DB2 policy.

PLAN=

Specify the plan name against which an associated action is to be applied. The PLAN= statement must precede its associated ACTION= statement. If you are specifying the complete plan name, it must be one to eight characters. You may use the wildcard character (*) with the initial characters of the plan name to be matched. The following examples illustrate the use of the wildcard character to match plan names.

PLAN=ACC* indicates that Job Optimizer for DB2 is to perform the associated action on all plans that start with ACC.

PLAN=* indicates that Job Optimizer for DB2 is to perform the associated action on all plans.

PLAN=ABCDEindicates that Job Optimizer for DB2 is to perform the associated action on one plan that is named ABCDE.

TABLE=

Specify the table name against which an associated action is to be applied. The TABLE= statement must precede its associated ACTION= statement. You must specify an actual table name. You should not specify aliases, synonyms, or views. However, Job Optimizer for DB2 will process any SQL statements that contain aliases, synonyms, or views to find the underlying table(s). The associated action will then be applied to those underlying tables that match the table in the DB2 policy statement.

If the specified table is involved in any referential integrity relationships, the Job Optimizer for DB2 will *not* apply the associated action to any dependent or parent tables. For more information on Job Optimizer for DB2 and referential integrity, see Appendix D, “Job Optimizer for DB2 and SQL Statements.”

If you specify a qualified table name, it must have a 1-to 8-character qualifier followed by a period, which is then followed by a 1-to 18-character table name. Implicit qualification and wildcard character (*) usage is allowed. If you do not specify a period, an implicit qualifier of * is assumed, which means that any qualifier and the value specified is assumed to be the actual table name. You can use the wildcard character in either the qualifier name or the actual table name by specifying the initial characters to be matched followed by an asterisk.

Table 13-1 illustrates the use of the wildcard character to match table names. The TABLE column is the value you specify in the policy. The Qualifier Name and Table Name columns are the values interpreted by Job Optimizer for DB2 for its matching. The Example Table shows a possible table Job Optimizer for DB2 might find when it tries to find tables that match its interpreted values of the TABLE= option statement.

Table 13-1 TABLE= Value Examples

TABLE	Qualifier Name	Table Name	Example Table
X*.*	anything starting with X	anything	XACCT.EMPLL
X*	anything (implicit)	anything starting with X	XPROJ_TABL
X.*	X	anything	X.PROJ_TABL
*.XYZ	anything	XYZ	XYZ
.	anything	anything	EMP_TABL
*	anything (implicit)	anything	DEPT_TABL
ABC	anything (implicit)	aBC	ABC
A.B	A	B	A.B
EMP*	anything (implicit)	anything starting with EMP	EMP_TABL

BMC Software, Inc., Confidential and Proprietary Information

ACTION=

Specify the override action that is to be applied to the PLAN= or TABLE= statement that precedes it.

The following are the DB2 policy action options:

- SERIALIZE - specifies that a step that will access the specified plan or table will always run alone, just as it would have without Job Optimizer. The SERIALIZE action specified against a table rolls up to the plan level for the plan in which the table exists.
- SHARE - specifies that access to the plan or table will be treated as read-only and to allow appropriate parallel access with other steps that are read-only. The SHARE action specified against a plan cascades down to the table level and applies to all tables in the plan.
- NOSHARE - indicates that access to the plan or table be treated as update, and to prevent parallel access with other steps using the plan or table. The NOSHARE action that is specified against a plan cascades down to the table level and applies to all tables in the plan.
- EXCLUDE - this action parameter allows for plans to be specifically excluded from DB2 analysis.

VERSIONS=

The limit on the number of versions of a plan or package that Job Optimizer for DB2 will examine.

SSID=

Specify DB2 SSID to be excluded from a data sharing group that is defined as a DB2 Group from job optimization.

EXCLUDE=

Specify “Y” (for yes) to exclude an entire DB2 Group from job optimization.

Note: The DB2 Group may be a DB2 data sharing group or a single DB2 subsystem.

TIME_LIMIT=

Specifies the amount of elapsed time spent in gathering DB2 constraint information. The default is 5 minutes. The values can range from 1–60 minutes. If the time limit is exceeded during DB2 analysis, a DB2 constraint is applied.

CHECK_INTERVAL=

The Check Interval setting which allows for a period of time to elapse between checking the DB2 catalog for any plan or package changes. The default is derived from the time limit setting, but can be overridden by the CHECK_INTERVAL command. Values can range from 1–60 minutes.

Example DB2 Policy Created by the ISPF Dialog

Figure 8-1 is a sample of the DB2 policy that is created by the ISPF dialog. An asterisk in column 1 indicates a comment-only statement.

BMC Software, Inc., Confidential and Proprietary Information

Figure 13-1 Example DB2 Policy

```
*DB2POLICY
*-------------------------------------------------------------------------
*GLOBAL OPTIONS (note they are comment only, so defaults are used
*-------------------------------------------------------------------------
*GROUP_AUTHID= (BATCH ACCELERATOR for DB2 uses default INSTALL SYSADM)
*GROUP_PLAN= (BATCH ACCELERATOR for DB2 uses default MBOPLAN)
*-------------------------------------------------------------------------
*END OF GLOBAL OPTIONS
*-------------------------------------------------------------------------
*DB2GROUP OPTIONS FOR DB2 ATTACH GROUP DSNDBT
*-------------------------------------------------------------------------
DB2GROUP=DSNDBT
TABLE=CHECKPT*       \ all tables starting with CHECKPT in this DB2Group
should have share
ACTION=SHARE         \ access, even for insert, update, and delete
operations
*
PLAN=DSN*            \ plans starting with DSN to be share (all tables
ACTION=SHARE         \  accessed are to be marked share, regardless of
access)
TABLE=ENQ_TABLE      \ in all DSN* plans, the ENQ_TABLE will given update
access
ACTION=NOSHARE       \ even when the plan accesses the table read-only
*
*The order of the policy definitions determines which policy takes
precedence in possible conflicts. The definition for PLAN=DSN*,
ACTION=SHARE is overridden for by the more specific TABLE=ENQ_TABLE
ACTION=SERIALIZE. If plan
DSNACCT accesses the ENQ_TABLE, the action will be SERIALIZE .
*-------------------------------------------------------------------------
*END OF DB2GROUP OPTIONS FOR DB2 ATTACH GROUP DSNDBT
*-------------------------------------------------------------------------
*-------------------------------------------------------------------------
*DB2GROUP OPTIONS FOR DB2 ATTACH GROUP DSNDDB
*-------------------------------------------------------------------------
DB2GROUP=DSNDDB
SSID=DDB1                 \__exclude member DDB1 from the available
subsystems
ACTION=EXCLUDE            / within the Data Sharing Group receiving work
from MBO
TABLE=RESTART*            \__all tables starting with RESTART should have
share
ACTION=SHARE               / access, even for insert, update, and delete
operations
TABLE=ENQ*                 \__all plans accessing tables starting with ENQ
should
ACTION=SERIALIZE            / be serialized even when the plan is read-only
access
*-------------------------------------------------------------------------
```

```
*END OF DB2GROUP OPTIONS FOR DB2 ATTACH GROUP DSNDDB
```

Special Considerations

Prior to your implementation of Job Optimizer for DB2, you should consider the following action items:

System Compatibility

Job Optimizer for DB2 allows steps in a job to be run on a system different from the one that initiated it. In the case of a DB2 step, the step runs on a system with an active DB2 data sharing member. Ensure that all other supporting software is available.

LINKLIST compatibilityMust be the same on all systems

Software compatibilityIBM and third party software must be on the same level on a compatible level release level. The IBM Language Environment is an example of a product that may not be compatible among releases.

Subsystem compatibilitySubsystems may need to be available on all systems which may have DB2 steps running on them.

Static Structure/Access Count

Job Optimizer lets you to specify the Structure Change Count Option in the Job Policy Global Options to indicate the number of times that the data access patterns for a job step must remain static before data constraints can be relieved. This structure change count works differently in Job Optimizer for DB2. The trust factor will be reset for DB2 steps only when the PLAN NAME or PROGRAM NAME is changed. For a DB2 step, this also indicates the stability of the plan or plans accessed by the step based on the HISTORY data set. The step must access the same plans this number of times before it can be considered for parallel execution.

For more information on the Structure Change Count, see “Structure Change Count” on page 4-25.

Number of Concurrent Batch Users

DB2 users should consider increasing the maximum number of concurrent batch users. Job Optimizer for DB2 executes job steps in parallel that were previously executed serially, and shifts steps from other systems. Increase the number of concurrent batch users by at least the maximum number of DB2 steps in a job or the MAINVIEW Batch Optimizer number of initiators option.

For more information on Static Structure/Access Count, see Chapter 3, “Implementing Job Optimizer.”

DB2 Call Attachment Facility Applications

For DB2 applications using the Call Attachment Facility, Job Optimizer for DB2 may distribute the job step to other MVS images within the SYSPLEX, where a suitable DB2 member is available, when using the DB2 group attach name. If a specific DB2 SSID is used, the job step is limited to the originating, or Home, MVS image.

Note: When changing from a DB2 group attach name to a specific DB2 SSID, after Job Optimizer for DB2 has recorded history, the job step may be inadvertently distributed to a MVS image where that DB2 SSID is not operational.

RRSAF Plan Name ?RRSAF

RRSAF plan names that begin with the question mark character (*?)* will not be processed by Job Optimizer for DB2.

Indexing the DB2 Catalog

In addition to these other considerations, you can also improve MAINVIEW Batch Optimizer for DB2 performance by adding indexes to the DB2 catalog tables. For more information, see install library member BSADB2IX in the SAMPLIB data set.

Creating Policies for Job Optimizer for DB2

This section describes dialog panels and how you can use them to create a DB2 policy.

For Job Optimizer for DB2 to consider taking action on a batch job, the base Job Optimizer product must first select the job for processing. Job Optimizer can take action on a batch by ensuring that the current Job Optimizer Job Policy contains a Selection Criteria (SELDEF) entry, which will cause Job Optimizer to take action on the job. For more information about the syntax and criteria of a Job Policy Selection Definition entry, see Chapter 4, “Defining Job Policies.”

The Job Optimizer’s Selection Definition will cause action to be taken on the job, but to provide maximum integrity and flexibility, a separate policy specifically intended for DB2 batch steps is required. This DB2 Policy, while the primary selection criteria may be the Jobname, has the capability of specifying other criteria, such as DB2 PLAN and DB2 TABLE name, to allow the decision to split to be made at the STEP level.

An example of an DB2 Policy can be found in the Sample Plex dataset member DB2POL00. The Sample Plex dataset was downloaded to your system DASD from product installation tape dataset BMC.BSS.SAMPLIB.

DB2 policies may be created, edited and viewed, using the MAINVIEW Batch Optimizer dialog. The creating DB2 policy task descriptions in this chapter assume that you are familiar with the dialog’s initial panel sequence and basic navigation techniques.

Help is available online for all panels, pop-ups, and fields while creating DB2 policy definitions.

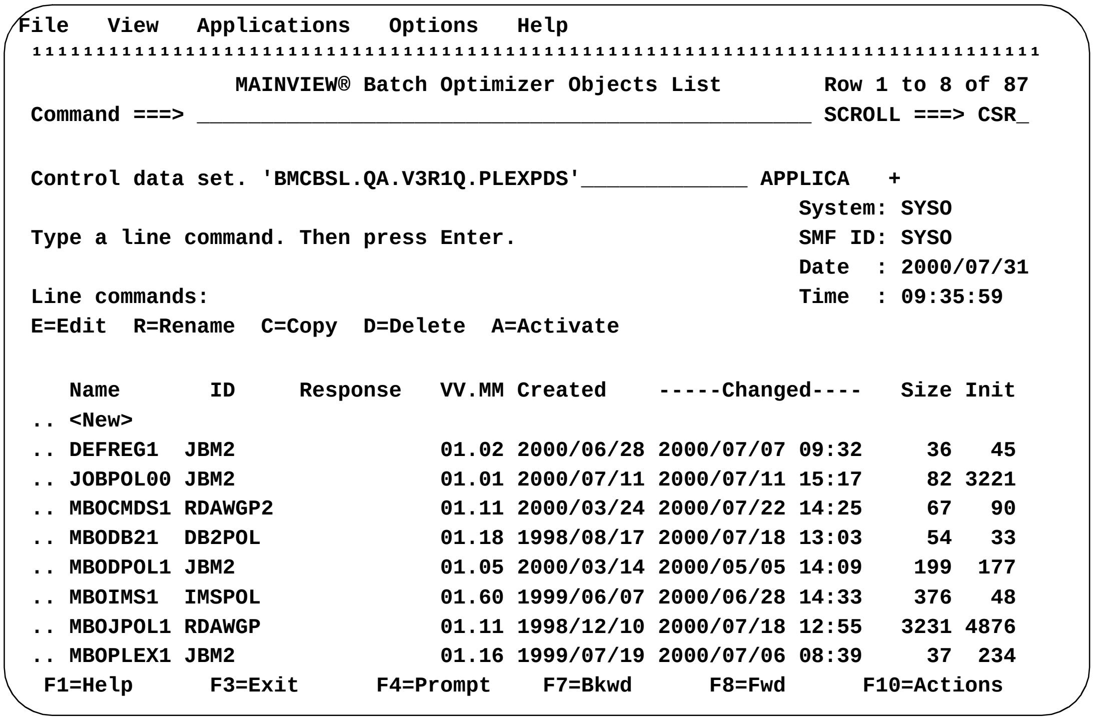
Editing or Viewing a DB2 Policy

You can review or modify the information saved for a DB2 policy, including the actions associated with each group in the DB2 Policy.

To edit or view information for a DB2 policy in the control data set, complete the following steps:

Step 1 Access the MAINVIEW Batch Optimizer Objects List panel (Figure 13-2).

Figure 13-2 MAINVIEW Batch Optimizer Objects List Panel



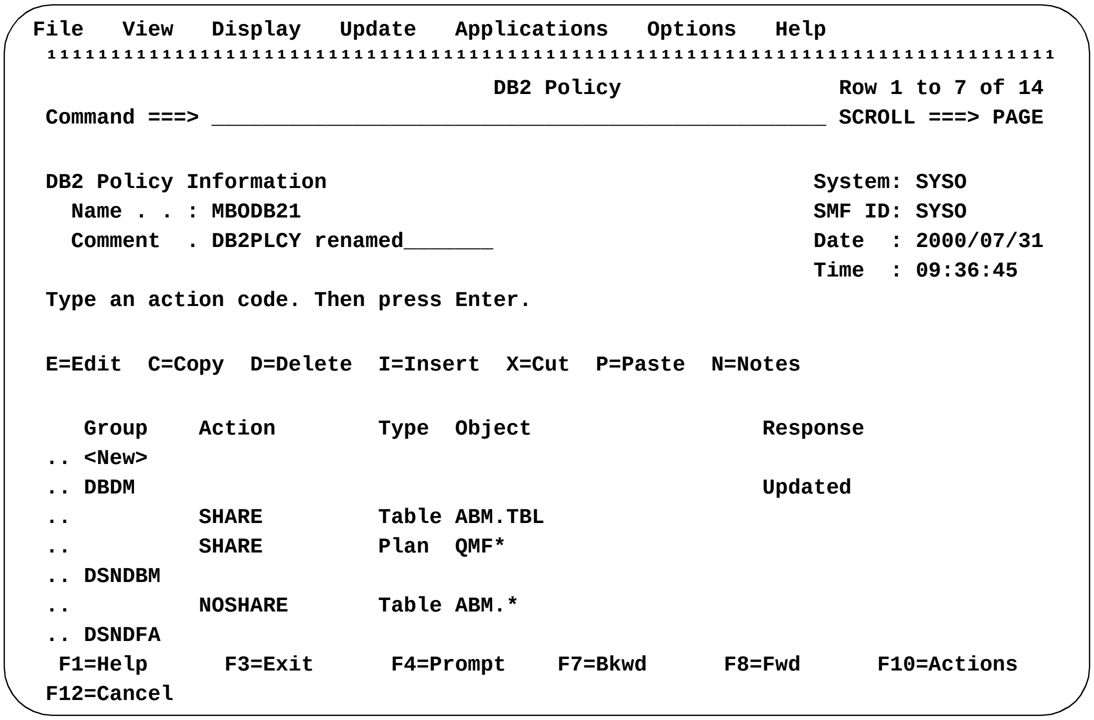
```
File   View   Applications   Options   Help
 1111111111111111111111111111111111111111111111111111111111111111111111111111111
                MAINVIEW® Batch Optimizer Objects List        Row 1 to 8 of 87
Command ===> ________________________________________________ SCROLL ===> CSR_

Control data set. 'BMCBSL.QA.V3R1Q.PLEXPDS'_____________ APPLICA   +
                                                            System: SYSO
Type a line command. Then press Enter.                      SMF ID: SYSO
                                                            Date  : 2000/07/31
Line commands:                                              Time  : 09:35:59
E=Edit  R=Rename  C=Copy  D=Delete  A=Activate

   Name       ID     Response   VV.MM Created    -----Changed----   Size Init
.. <New>
.. DEFREG1  JBM2                01.02 2000/06/28 2000/07/07 09:32     36   45
.. JOBPOL00 JBM2                01.01 2000/07/11 2000/07/11 15:17     82 3221
.. MBOCMDS1 RDAWGP2             01.11 2000/03/24 2000/07/22 14:25     67   90
.. MBODB21  DB2POL              01.18 1998/08/17 2000/07/18 13:03     54   33
.. MBODPOL1 JBM2                01.05 2000/03/14 2000/05/05 14:09    199  177
.. MBOIMS1  IMSPOL              01.60 1999/06/07 2000/06/28 14:33    376   48
.. MBOJPOL1 RDAWGP              01.11 1998/12/10 2000/07/18 12:55   3231 4876
.. MBOPLEX1 JBM2                01.16 1999/07/19 2000/07/06 08:39     37  234
 F1=Help      F3=Exit      F4=Prompt    F7=Bkwd      F8=Fwd      F10=Actions
```

Step 2 Position the cursor to the left of your choice of DB2 Policy. Type **E** (Edit), and press **Enter**. The dialog displays the DB2 Policy panel.

The DB2 Policy panel displays a list of existing DB2 Groups and actions. A DB2 Group defines a DB2 data sharing group name or a DB2 subsystem id. Each DB2 Group definition may be followed by one or more action definitions. The action definitions that follow each DB2 Group define overrides for DB2 Objects (PLAN/TABLE) for that DB2 Group. Figure 13-3 is an example of DB2 Groups followed by actions for each group displayed on the DB2 Policy panel.

Figure 13-3 shows an example of the DB2 Policy Definition panel.

Figure 13-3 DB2 Policy Definition Panel



```
File   View   Display   Update   Applications   Options   Help
 ıııııııııııııııııııııııııııııııııııııııııııııııııııııııııııııııııııııııııııııı
                                    DB2 Policy                 Row 1 to 7 of 14
 Command ===> ________________________________________________ SCROLL ===> PAGE

 DB2 Policy Information                                      System: SYSO
   Name . . : MBODB21                                        SMF ID: SYSO
   Comment  . DB2PLCY renamed_______                         Date  : 2000/07/31
                                                             Time  : 09:36:45
 Type an action code. Then press Enter.

 E=Edit  C=Copy  D=Delete  I=Insert  X=Cut  P=Paste  N=Notes

    Group    Action        Type  Object                  Response
 .. <New>
 .. DBDM                                                 Updated
 ..          SHARE         Table ABM.TBL
 ..          SHARE         Plan  QMF*
 .. DSNDBM
 ..          NOSHARE       Table ABM.*
 .. DSNDFA
  F1=Help      F3=Exit      F4=Prompt    F7=Bkwd      F8=Fwd      F10=Actions
 F12=Cancel
```

Step 3 To make changes to the global options for the DB2 policy; for details see “Editing or Viewing DB2 Policy Global Options” on page 13-24.

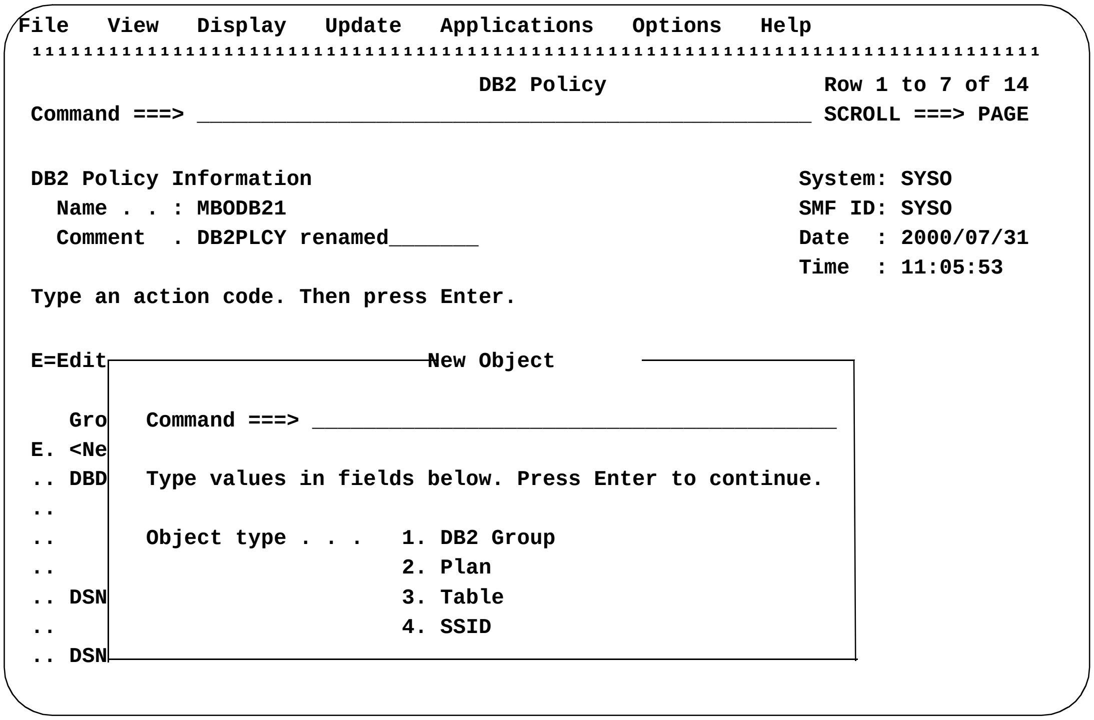
Step 4 To make changes to the DB2 Group or Action definition on the DB2 Policy panel. For details, see “Editing or Viewing a DB2 Policy GROUP or ACTION Definition” on page 13-26.

Step 5 You add new DB2 Group definition by doing the following:

Note: The order of DB2 policy definitions affects Job Optimizer for DB2 operation.

- To add a new DB2 Group definition. Type **E** (EDIT) next to **<NEW>**. The dialog displays the New Object selection pop-up window as shown in Figure 13-4 on page 13-19.

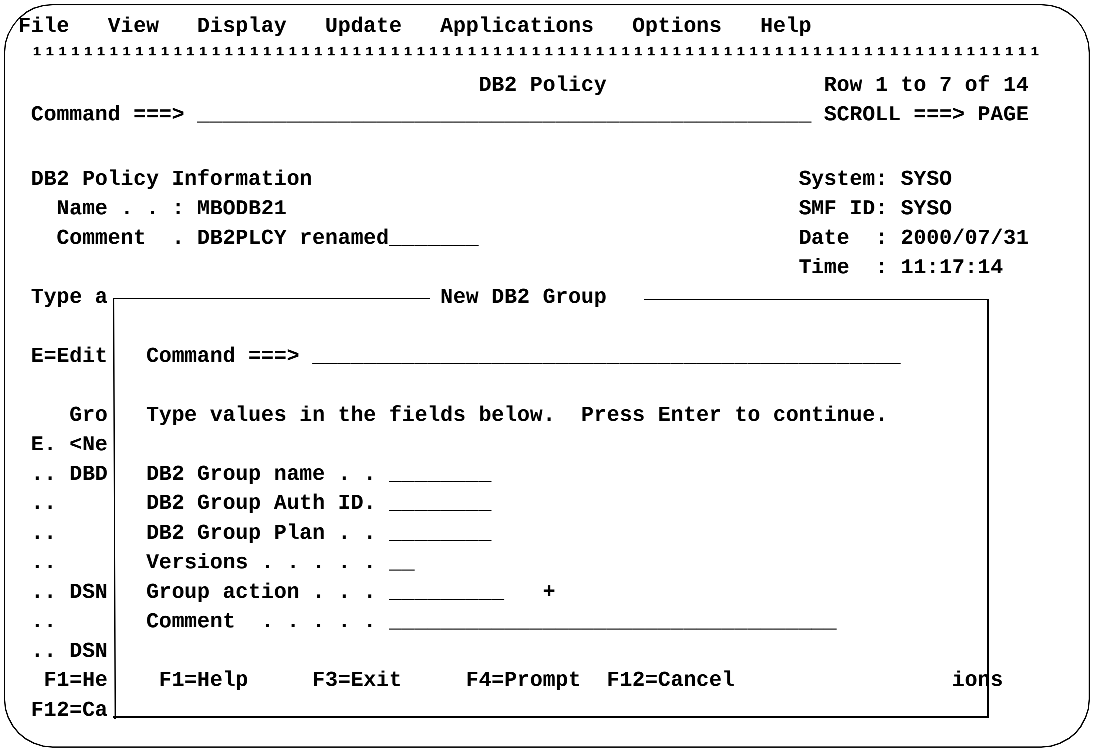
Figure 13-4 New Object Pop Up Panel



```
File   View   Display   Update   Applications   Options   Help
 11111111111111111111111111111111111111111111111111111111111111111111111111111111
                                  DB2 Policy                 Row 1 to 7 of 14
Command ===> ________________________________________________ SCROLL ===> PAGE

DB2 Policy Information                                     System: SYSO
  Name . . : MBODB21                                       SMF ID: SYSO
  Comment  . DB2PLCY renamed_______                        Date  : 2000/07/31
                                                           Time  : 11:05:53
Type an action code. Then press Enter.

E=Edit                        New Object

   Gro   Command ===> _________________________________________
E. <Ne
.. DBD   Type values in fields below. Press Enter to continue.
..
..       Object type . . .   1. DB2 Group
..                           2. Plan
.. DSN                       3. Table
..                           4. SSID
.. DSN
```

Select option 1 for DB2 group from the New Object pop-up panel Shown in Figure 13-4 on page 13-19. The New DB2 Group pop-up panel will be displayed as shown in Figure 13-5.

Figure 13-5 New DB2 Group Pop Up Panel



```
File   View   Display   Update   Applications   Options   Help
11111111111111111111111111111111111111111111111111111111111111111111111111111111
                                   DB2 Policy                Row 1 to 7 of 14
Command ===> ________________________________________________ SCROLL ===> PAGE

DB2 Policy Information                                     System: SYSO
  Name . . : MBODB21                                       SMF ID: SYSO
  Comment  . DB2PLCY renamed_______                        Date  : 2000/07/31
                                                           Time  : 11:17:14
Type a                          New DB2 Group

E=Edit   Command ===> ______________________________________________

   Gro   Type values in the fields below.  Press Enter to continue.
E. <Ne
.. DBD   DB2 Group name . . ________
..       DB2 Group Auth ID. ________
..       DB2 Group Plan . . ________
..       Versions . . . . . __
.. DSN   Group action . . . _________   +
..       Comment  . . . . . ___________________________________
.. DSN
 F1=He    F1=Help     F3=Exit     F4=Prompt  F12=Cancel                ions
F12=Ca
```

- The New DB2 Group pop-up panel allows you to enter the following information:

  — The DB2 Group Name is a DB2 data-sharing group or a DB2 subsystem ID.

  — The Group AUTHID is the DB2 authorization ID that will be used by the Job Optimizer of DB2 Plan.

  — The Group PLAN is the DB2 plan that is used by Job Optimizer for DB2.

  — Number of versions of a plan or package that Job Optimizer for DB2 will examine.

  — Exclude a DB2 group for job optimization.

For more information on the fields defined on the New DB2 Group panel, see “DB2 Policy Information” on page 13-7.

After making changes to the New DB2 Group pop-up panel, press **Enter**.

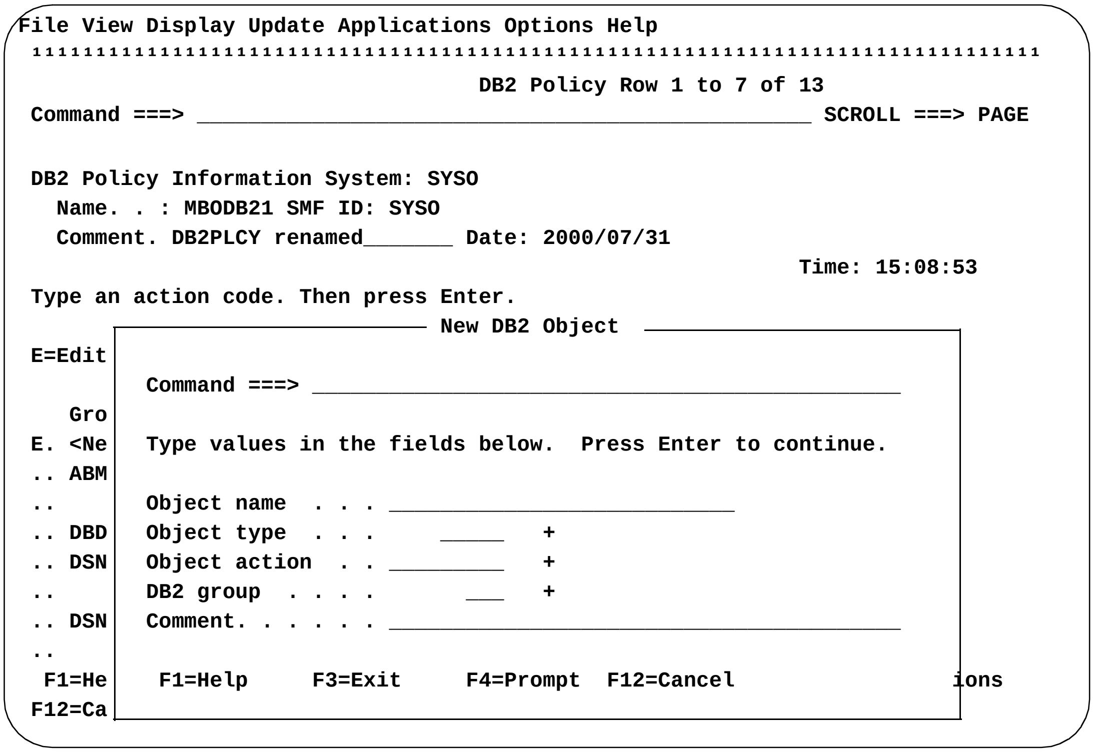
Step 6 Add a new Action definition for a DB2 Group by completing the following steps:

Note: The order of DB2 policy definitions affects Job Optimizer for DB2 operation.

6.A To add a new Action definition for a DB2 Group. Type **E** (EDIT) next to **<NEW>**. The dialog displays the New Object Selection pop-up panel as shown in Figure 13-4 on page 13-19.

6.B Select option 2 for PLAN or option 3 for TABLE from the New Object pop-up panel. The New DB2 Object pop-up panel will be displayed as shown in Figure 13-6.

Figure 13-6 New DB2 Object Pop-Up Panel



```
File View Display Update Applications Options Help
 1111111111111111111111111111111111111111111111111111111111111111111111111111111
                                   DB2 Policy Row 1 to 7 of 13
Command ===> ________________________________________________ SCROLL ===> PAGE

DB2 Policy Information System: SYSO
  Name. . : MBODB21 SMF ID: SYSO
  Comment. DB2PLCY renamed_______ Date: 2000/07/31
                                                            Time: 15:08:53
Type an action code. Then press Enter.
       ------------------------ New DB2 Object  ---------------------------
E=Edit |
       |  Command ===> ______________________________________________
   Gro |
E. <Ne |  Type values in the fields below.  Press Enter to continue.
.. ABM |
..     |  Object name  . . . ___________________________
.. DBD |  Object type  . . .     _____   +
.. DSN |  Object action  . . _________   +
..     |  DB2 group  . . . .       ___   +
.. DSN |  Comment. . . . . . ________________________________________
..     |
 F1=He |   F1=Help     F3=Exit     F4=Prompt  F12=Cancel                 |ions
F12=Ca |__________________________________________________________________|
```

- The New DB2 Object pop-up panel lets you to enter the following information:

  - DB2 Object name and type for a DB2 PLAN or TABLE for which normal Job Optimizer for DB2 processing will be overridden.

  - Specify the action for the PLAN or TABLE that is to be performed by Job Optimizer for DB2 such as SERIALIZE, see “DB2 Policy Information” on page 13-7.

  After making changes to the New DB2 Object pop-up panel then press **Enter**.

The following are shortcuts for modifying the DB2 Policy:

- To insert (similar to add, but in your choice of position) a new DB2 Group definition or action, decide which DB2 Group definition or action you want the new definition to be placed after. Position the cursor in the action.

  Enter field to the left of your choice of DB2 Group definition or action, type **I** (Insert), and press **Enter**. The NEW OBJECT selection pop-up panel as shown in Figure 13-4 on page 13-19 is displayed. Select option 1 to insert a new DB2 Group definition. The New DB2 Group panel is displayed as shown in Figure 13-5 on page 13-20. Select option 2 or 3 to insert a for a new action definition for a DB2 Group. The New DB2 Object panel is displayed as shown in Figure 13-6 on page 13-21.

- To copy (and paste) a DB2 Group definition or action, position the cursor in the action entry field to the left of your choice of DB2 Group definition or action. Type **C** (Copy), and press **Enter**. The dialog response is: Copied.

  Decide which DB2 Group or action definition you want the copied definition to be placed after. Position the Cursor in the action entry field to the left of you choice of DB2 Group or action definition. Type **P** (Paste), and Press **Enter**. The dialog moves the definition from the clipboard to your selected position.

- To move (cut and paste) a DB2 Group or action definition, position the cursor in the action entry field to the left of your choice of DB2 Group or action definition. Type **X** (Cut), and press **Enter**. The dialog response is: cut.

  Decide which DB2 Group or action definition you want the cut definition to be placed after. Position the cursor in the entry field to the left of your choice of DB2 Group or action definition. Type **P** (Paste), and press **Enter**. The dialog moves the definition from the clipboard to the list.

- To delete a DB2 Group or action definition, position the cursor in the action entry field to the left of your choice of DB2 Group definition. Type **D** (Delete), and press **Enter**. The dialog displays a Confirm Delete pop-up, which allows you to confirm the delete request before the DB2 Group or action definition is removed from the DB2 Policy.

Step 7 You can save your changes by using the following methods:

- Select the file pull-down choice 2 (Save DB2 Policy). The dialog saves the changed DB2 Group or action, and the DB2 Policy panel is displayed.

- Press **F3** (Exit). The dialog displays a Confirm Exit pop-up. Type **1** (Save policy changes to disk and exit), and press **Enter**.

BMC Software, Inc., Confidential and Proprietary Information

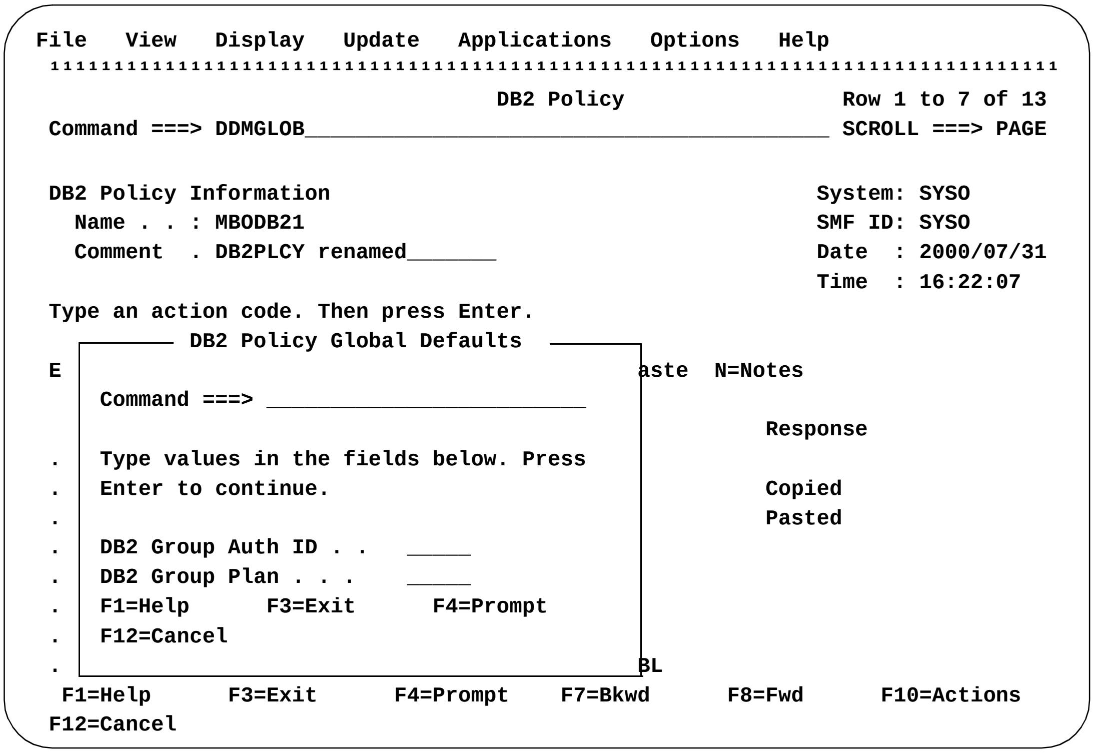
Editing or Viewing DB2 Policy Global Options

You can review or modify the global option values saved in a DB2 policy. To edit or view global option for a DB2 Policy in the control data set, complete the following steps:

Step 1 Access the DB2 Policy panel (Figure 13-3 on page 13-18). For information about accessing this panel, see "Editing or Viewing a DB2 Policy" on page 13-17.

Step 2 Position the cursor on the action bar Options option, and press **Enter**. Select option 6 from the selection pull-down for DB2 policy default options. The DB2 Policy Global Defaults pop-up panel in Figure 13-7 is displayed, as shown in Figure 13-7.

Figure 13-7 New DB2 Policy Global Defaults Pop-Up Panel



```
File   View   Display   Update   Applications   Options   Help
 ıııııııııııııııııııııııııııııııııııııııııııııııııııııııııııııııııııııııııııııı
                                  DB2 Policy                Row 1 to 7 of 13
 Command ===> DDMGLOB_________________________________________ SCROLL ===> PAGE

 DB2 Policy Information                                    System: SYSO
   Name . . : MBODB21                                      SMF ID: SYSO
   Comment  . DB2PLCY renamed_______                       Date  : 2000/07/31
                                                           Time  : 16:22:07
 Type an action code. Then press Enter.
      _______ DB2 Policy Global Defaults _______
 E  |                                           |aste  N=Notes
    |  Command ===> _________________________   |
    |                                           |       Response
 .  |  Type values in the fields below. Press   |
 .  |  Enter to continue.                       |       Copied
 .  |                                           |       Pasted
 .  |  DB2 Group Auth ID . .   _____            |
 .  |  DB2 Group Plan . . .    _____            |
 .  |  F1=Help      F3=Exit      F4=Prompt      |
 .  |  F12=Cancel                               |
 .  |___________________________________________BL
  F1=Help      F3=Exit      F4=Prompt    F7=Bkwd      F8=Fwd      F10=Actions
 F12=Cancel
```

The DB2 Policy Global Defaults panel, lets you to specify or change the values that serve as the default field values for the DB2 policy, if none are specified for the DB2 Group definition. AUTHORIZATION ID, PLAN, and VERSIONS are the only options that can be specified globally. The AUTHORIZATION ID is the DB2 authorization ID that will be used by Job Optimizer for DB2. The PLAN is DB2 plan that will be used by Job Optimizer for DB2. If the AUTHORIZATION ID field is left blank and no authorization id was specified when the DB2 GROUP definition was created, the default value is the install SYSADM. If the PLAN field is left blank, the default value is MBOPLAN.

Note: Note: The Global options are used only when no values for the GROUP AUTHID and GROUP PLAN are specified for a DB2 Group definition. DB2 Group definition options override the GLOBAL options.

Step 3 You can make changes to an entry fields on the DB2 Policy Global Defaults panel. Position the cursor in the field and type a new value. Blank out any characters that remain from the previously displayed value. Repeat this step for each field that you edit.

Step 4 Press Enter to retain you changes within the dialog, or press F12 (Cancel) to discard your changes. The dialog displays the DB2 Policy panel, where you can save changes to the control data set. To save changes, select the File pull-down choice 2 (Save DB2 Policy). The dialog saves the changes in the DB2 policy and redisplays the DB2 Policy panel.

Editing or Viewing a DB2 Policy GROUP or ACTION Definition

You can review or modify the information saved for a DB2 policy group or action definition. To edit or view a DB2 GROUP or ACTION definition in a DB2 Policy, complete the following steps:

Step 1 Access the DB2 Policy Panel (Figure 13-3 on page 13-18). See “Editing or Viewing a DB2 Policy” on page 13-17.

Step 2 Position the cursor in the action entry field to the left of your choice of DB2 Group or action. Type **E** (Edit), and press **Enter**. If you are editing a DB2 Group, the NEW DB2 GROUP pop-up panel is displayed as in Figure 13-5 on page 13-20. If you are editing an ACTION, the New DB2 Object pop-up panel as shown in Figure 13-6 on page 13-21.

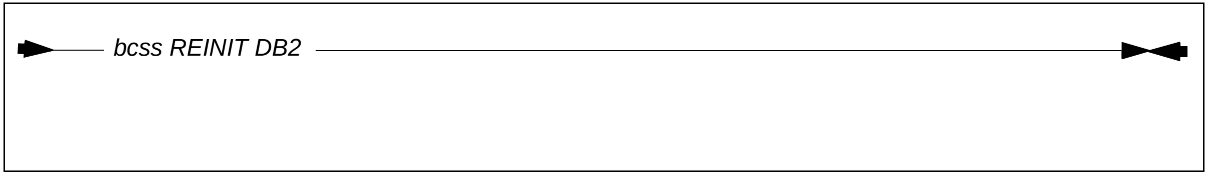
Step 3 Modify fields of your choice, press **Enter**.

Step 4 To save changes, select File pull-down choice 2 (Save DB2 Policy). The dialog saves the changes in the DB2 policy and the DB2 Policy panel is displayed.

Activating Job Optimizer for DB2 Processing

To activate Job Optimizer, issue the following command from the MVS console:

Figure 13-8 Activating Job Optimizer Processing



```
bcss REINIT DB2
```

bcss is the ID of the MAINVIEW Batch Optimizer subsystem that is active on the MVS image.

Successful initialization will result in a BMC193200I message being issued.

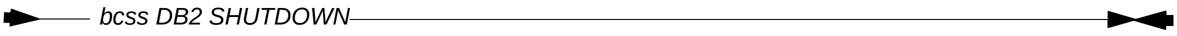
Note: As an alternative to the above step, you can activate Job Optimizer for DB2 processing by adding the REINIT DB2 command to the BatchPlex Global MAINVIEW Batch Optimizer Subsystems Options Commands member. If specified, you can also add the MVS Image MAINVIEW Batch Optimizer Subsystem Options Commands member.

BMC Software, Inc., Confidential and Proprietary Information

Terminating Job Optimizer for DB2

To terminate Job Optimizer for DB2 processing for a particular job, issue the following command from the MVS console:

Figure 13-9 Terminating Job Optimizer for DB2 Processing



bcss DB2 SHUTDOWN

bcss is the ID of the MAINVIEW Batch Optimizer subsystem that is active on the MVS image.

Successful terminations will result in a BMC193201I message being issued.

Activating a DB2 Policy

To activate an DB2 policy, issue the following command from the MVS console:

Figure 13-10 Activating an DB2 Policy

```
►►── bcss DB2 POLICY ACTIVATE policyname ──────────────────────►
```

bcss is the ID of the MAINVIEW Batch Optimizer subsystem that is active on the MVS image.

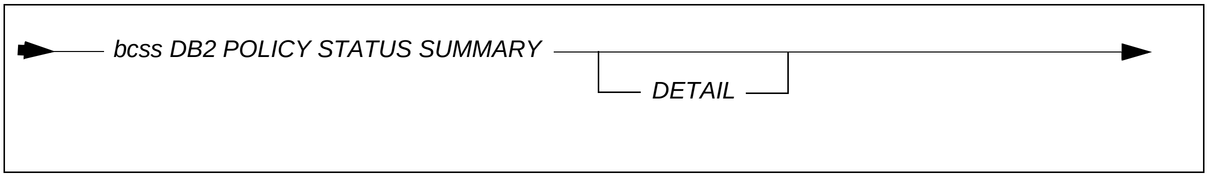
Note: As an alternative to the above step, from the User Interface’s MAINVIEW Batch Optimizer Objects List panel, (panel BSSP00IP), type an “A” (for Activate) in the command column on the line which contains the name of the DB2 policy you wish to activate.

Successful activation will result in a BMC193001I message being issued.

Displaying the Active DB2 Policy

To display the name of the active DB2 policy, issue the following command from the MVS console:

Figure 13-11 Displaying the Active DB2 Policy



bcss is the ID of the MAINVIEW Batch Optimizer subsystem that is active on the MVS image.

Message BMC193012I will be issued indicating the currently active DB2 POLICY name. If there was not an active DB2 POLICY, a BMC193013I message will be issued.

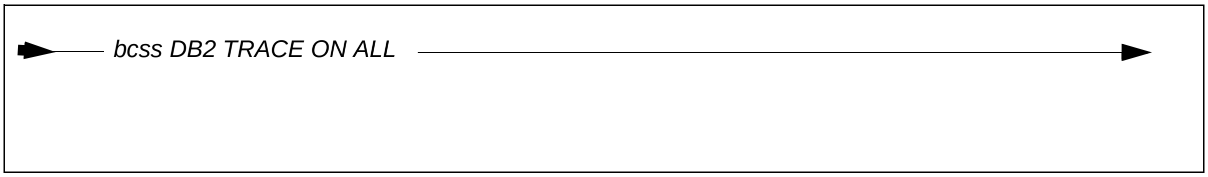
For DETAIL, message BMC193022I will be issued.

Activating Tracing for Job Optimizer for DB2 Modules

Tracing of Job Optimizer for DB2 modules should be done *only* when requested by BMC Software support staff. Tracing for the base Job Optimizer component *must* be ACTIVE.

To start tracing Job Optimizer for DB2 modules, issue the following command from the MVS console:

Figure 13-12 Activating Tracing for DB2 Modules



```
bcss DB2 TRACE ON ALL
```

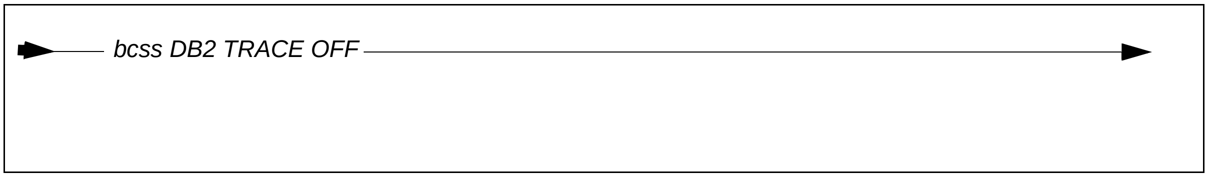
bcss is the ID of the MAINVIEW Batch Optimizer subsystem that is active on the MVS image.

Successful activation will result in message BMC193017I being issued indicating that tracing is On.

Terminating Tracing for Job Optimizer for DB2 Modules

To terminate tracing of Job Optimizer for DB2 modules, issue the following command from the MVS console:

Figure 13-13 Terminating Tracing for DB2 Modules



```
bcss DB2 TRACE OFF
```

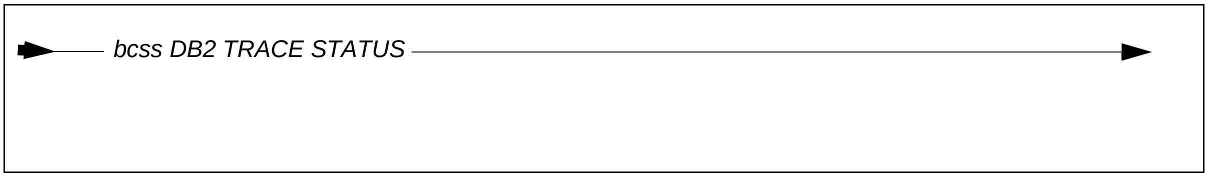
bcss is the ID of the MAINVIEW Batch Optimizer subsystem that is active on the MVS image.

Successful activation will result in a BMC193017I message being issued indicating that tracing is off.

Displaying Status of Job Optimizer for DB2 Tracing

To display the status of Job Optimizer for DB2 tracing, issue the following command from the MVS console:

Figure 13-14 Displaying the Status of DB2 Tracing



bcss DB2 TRACE STATUS

bcss is the ID of the MAINVIEW Batch Optimizer subsystem that is active on the MVS image.

Chapter 14 Job Optimizer for IMS

This chapter explains what Job Optimizer for IMS is, what it does, and how it works. It also explains system requirements. This chapter discusses the following topics:

BMC Software, Inc., Confidential and Proprietary Information

Overview

Job Optimizer for IMS provides support for batch workloads that involve IMS DLI, DBB, and BMP region processing. Job Optimizer for IMS identifies and supports job steps that reference IMS databases, thereby allowing parallelism for these job steps. Job Optimizer for IMS provides workload management and optimization for IMS jobs in a sysplex and non-sysplex environment.

Support for Batch Workloads that Involve IMS

MAINVIEW Batch Optimizer invokes Job Optimizer for IMS when it encounters a job step that involves IMS processing. Job Optimizer for IMS works with Job Optimizer to provide similar benefits for IMS job steps to those that Job Optimizer provides for non-DBMS steps.

Parallelism for Job Steps that Reference IMS

Job Optimizer for IMS analyzes IMS DLI, DBB and BMP steps and identifies the IMS dependencies of those job steps. The result is determination of which IMS job steps can run in parallel on the same or other MVS images. Job Optimizer for IMS also determines which images are eligible to run each step. For example, parallel processing of steps that use IMS could occur on images that are capable of sharing the same IMS databases.

Workload Management and Optimization for IMS Jobs

Job Optimizer for IMS provides workload management and optimization for IMS jobs by distributing job steps across multiple images that are capable of sharing IMS databases for parallel processing. Within a sysplex, processing capabilities can vary, so the ability to target a job step to an appropriate image helps balance the workload.

How Job Optimizer for IMS Works

Job Optimizer for IMS provides support for batch workloads that involve IMS DLI, DBB, and BMP region processing. Job Optimizer for IMS accomplishes this task for MAINVIEW Batch Optimizer customers by providing the following features

- user-specified processing policies
- history data set reflecting database access and job execution information
- constraint analysis
- job analysis

User-Specified Processing Policies

Job Optimizer for IMS lets you specify preferences for IMS processing policies. This feature is available through the ISPF interface. Policies are available at JOBNAME, PROGRAM NAME (PGM) and PROGRAM SPECIFICATION BLOCK (PSB) levels. You can specify the criteria necessary before Job Optimizer for IMS provides performance processing for a batch job. You can also define what action Job Optimizer for IMS should perform for the selected batch job.

History Data Set Reflecting Database Access and Job Execution Information

Job Optimizer for IMS records all statistics and information about database access and job execution in the same VSAM history data set that MAINVIEW Batch Optimizer uses for data capture and history recording. The database access information contained in the history record includes the ddnames and data set names referenced by the PSB and the access against them. The history record also contains job execution information such as databases accessed, the type of access (read or update), and execution parameters. Any change in the IMS region type, program name or PSB name will result in the step running serially on the home MVS image.

Constraint Analysis

Job Optimizer for IMS provides an interface with the MAINVIEW Batch Optimizer constraint analysis feature to relieve any constraints on the execution of IMS batch workloads. If there is a potential constraint for an IMS job step, the program compares current execution parameters to the data captured in the history data set. The reason for this comparison is to verify repeatability. Depending on other constraints, the job step may be able to run in parallel. However, Job Optimizer for IMS does not allow parallel processing for IMS job steps according to the following constraints:

- There is no action taken by Job Optimizer for IMS if the processing involves *any* IMS utility job steps. Subject to the normal constraints of MAINVIEW Batch Optimizer, the job steps are treated as normal MVS job steps.

- A step which contains IMS database updates cannot run in parallel with any other prior step that writes data that is used as input by the IMS update step.

- A step which contains IMS database updates cannot run in parallel with any other IMS step when the two steps have PSB’s that reference a common database.

- A step which updates IMS databases will not run in parallel with any other step if the IMS policy for this step indicates DLIREAD or DLIABEND. These steps may execute on other MVS images within the sysplex. To remove this constraint, the IMS policy must specify SPLIT for this step.

Job Analysis

Job Optimizer for IMS identifies the candidate jobs and recommends IMS batch workload execution changes, based on statistics and information about processing options gathered over time. The recommended IMS batch workload execution changes may include the following tasks

- redirecting IMS batch job steps to other processors in the sysplex

- allowing two or more job steps to run in parallel without application or JCL changes

- facilitating batch workload balancing across multiple systems

Splitting DLI/DBB Batch Steps with DBRC

When requesting that IMS DLI/DBB steps be capable of splitting by utilizing the SPLIT, DLIREAD or DLIABEND options of the “ACTION=” keyword in an IMS policy, and DBRC is being used, the user should be aware of the following:

- Databases accessed within a step NOT registered with DBRC are considered to have the highest Share Level (currently Share Level 3).
- Databases accessed within a step must have at least a DBRC Share Level of 1 or the step will not run concurrently with any other IMS step.
- Databases accessed within an IMS step must have at least a DBRC Share Level of 3 to execute on another MVS image within the BatchPlex.

If the IMS step is capable of running concurrently with other IMS steps, Job Optimizer for IMS will modify the DBRC signon name used from the JOBNAME to a tokenized value. This is done to prevent a duplicate signon error from DBRC, similar to the signon error you would receive if you ran the same job from different MVS images sharing the same RECON datasets at exactly the same time. As IMS steps that run concurrently are READ-ONLY in nature, there are, normally, no batch back-out implications. When Job Optimizer for IMS performs this change, a BMC194070I message indicating the modified value of the DBRC SSID is issued.

Splitting DLI/DBB Batch Steps with IRLM

When requesting that IMS DLI/DBB steps be capable of splitting by utilizing the SPLIT, DLIREAD or DLIABEND options of the “ACTION=” keyword in an IMS policy, and IRLM is being used, the user should be aware of the following conditions:

- IMS steps utilizing IRLM Release 2 (and above), will only be routed to other MVS images within the BatchPlex which have an IRLM with the “SCOPE=GLOBAL” parameter specified and share the CF Group Name with the IRLM on the MVS Image that the job was initiated on.
- IMS steps utilizing IRLM Release 1.5 will not be routed to other MVS images.
- If the step is capable of running concurrently with other IMS steps, Job Optimizer for IMS will modify the IRLM identify name used (with the exception of the first IMS step in the job), from the JOBNAME to a tokenized value. This name will be the same as used by DBRC.

Splitting DLI/DBB Batch Steps Accessing DB2 Databases via the Call Attach Facility

When requesting that IMS DLI/DBB steps be capable of splitting by utilizing the SPLIT, DLIREAD or DLIABEND options of the “ACTION=” keyword in an IMS policy, and DB2 databases are being accessed via the call attach facility, the user should be aware of the following.

- If Job Optimizer for IMS permits the step to split and/or allows step concurrency, these will not occur unless Job Optimizer for DB2 is active, has also selected this step for processing via its DB2 Policy process, and has determined that this step may run concurrently with other steps. For usage and considerations, see Chapter 12, “Job Optimizer Pipes Implementation Considerations.”

- If the step is capable of running concurrently with other IMS steps, Job Optimizer for IMS will modify the DB2 connection id used from the JOBNAME to a tokenized value. This name will be the same as used by DBRC. A BMC194071I message will be issued indicating the tokenized name that has been used. Steps which do not access DB2 using IMS services, i.e. steps not executing program DFSRRC00 with a region type of DLI or DBB, are ignored by Job Optimizer for IMS.

Splitting DLI/DBB Batch Steps without DBRC

When requesting that IMS DLI/DBB steps be capable of splitting by utilizing the SPLIT, DLIREAD or DLIABEND options of the “ACTION=” keyword in an IMS policy, and DBRC is not being used, the user should be aware of the following:

- Databases accessed within a step NOT utilizing DBRC are considered to have the highest Share Level (Share level 3).

- Not utilizing DBRC and IRLM indicates that database sharing management and lock management is not an issue with this step, therefore, no DBRC or IRLM considerations are made when Job Optimizer for IMS attempts to determine where and with what other steps this IMS step may execute.

Splitting IMS Database Update Steps

Job steps which perform updates to IMS databases may run in parallel lwith other steps with some restrictions. DLI, DBB, and BMP steps all adhere to the following:

- The selection criteria in the active IMS policy for the IMS update step must have an ACTION statement specifying SPLIT for the step to have the potential torun in parallel with other steps. Specifying DLIREAD or DLIABEND for the IMS database update step will result in the step having the potential to execute on other MVS images, but it will not run in parallel with any other steps.

- IMS database update steps which are not the reader of a writer/reader pair (i.e., receiving end of a data pipe) may execute in parallel with other non-IMS steps.

- IMS database update steps may run in parallel with other IMS database steps which read databases provided that the PSBs for each step do not have any databases in common.

- IMS database update steps may not run in parallel with other IMS update steps, regardless of whether they are in common databases or not.

Splitting BMP Steps

When requesting that IMS BMP steps be capable of splitting by utilizing the SPLIT, DLIREAD or DLIABEND options of the “ACTION=” keyword in an IMS policy, the user should be aware of the following:

- DLIREAD and DLIABEND are not functional options for the “ACTION=” keyword for BMP steps, and BMP steps which are selected by an IMS policy selection block with these parameters will be treated as if SPLIT was specified.

- BMP Steps will only be considered for execution on other MVS images within the BatchPlex when the IMSGROUP key word is specified in the IMS Policy selection block.

IMS Policy Processing

In order for Job Optimizer for IMS to consider taking action on a batch job, the job must first be selected by the base Job Optimizer product. This can be accomplished by ensuring that the current Job Optimizer JOB POLICY contains a SELECTION DEFINITION (SELDEF) entry which will cause Job Optimizer to take action on the job. For more information on the syntax and criteria of a Job Policy Selection Criteria entry, see the *MAINVIEW Batch Optimizer Installation Manual*.

The Job Optimizer's SELDEF will cause action to be taken on the job, but in order to provide maximum integrity and flexibility, a separate policy, specifically intended for IMS DLI batch and BMP steps is required. This IMS POLICY, while the primary selection criteria MAY be the Jobname, has the capability of specifying criteria, which allow the decision to split to be made at the STEP level.

An Iexample of an IMS Policy can be found in the Sample Plex dataset in member IMSPOL00. The Sample Plex dataset was downloaded to your system DASD from product installation tape dataset BMC.BSS.SAMPLIB.

IMS Policies may be created, edited and viewed, using the MAINVIEW Batch Optimizer Dialog.

IMS Policy Information

There are two user modifiable fields in the IMS Policy Information section. The COMMENT field allows the user to specify a 22 byte comment at the beginning for the policy member. The Default action field allows the user to define what action Job Optimizer for IMS will take when a job has been selected for processing, but no ACTION has been explicitly defined for that selection. The actions permitted and the implications of each are discussed in OPTIONS section.

Selection Criteria

The selection criterion for Job Optimizer for IMS consists of 3 fields. These 3 fields are Jobname, Program and PSB and these constitute a selection block. After a job has been selected for processing by the base Job Optimizer product and the Job Optimizer for IMS is active, selected information for each step of the job is passed to Job Optimizer for IMS, and is compared with the selection block to determine if the step should be considered for Job Optimizer for IMS processing. Each of the three fields are tested in sequence and if any field does not match, the step is NOT selected and the next selection block is used to test the step’s information. When all the selection blocks have been exhausted, and no match has been found, the next step in the job is used to conduct the selection test. If all steps have been processed, and no complete match is found, Job Optimizer for IMS will ignore the job.

In the case of EQ, for example, if the executing Jobname “EQUALS” the selection Jobname field, then perform the Program name test. If not, then perform the Jobname test specified by the next selection block.

When NE is used, if the executing Jobname “DOES NOT EQUAL” the selection Jobname field, then perform the Program name test. If not, then perform the Jobname test specified by the next selection block.

Each of the fields allows a 1 to 8 character name to be specified against which the comparisons are performed. A wildcard character, which is an asterix, may be specified at any position, and any characters following the wildcard character will be ignored. If the wildcard character is specified in the first position, all names will be considered a match for that particular field.

For example, if “JOBNAME EQ * ” was specified, all executing jobs would be considered a match, and the Program name test would be executed. If “JOBNAME EQ ABMCJOB*” was specified, all jobs with the name ABMCJOB**A** through ABMCJOB**9**, plus ABMCJOB_(blank) would be considered a match.

The Jobname field is compared with the executing Jobname. The Program field is compared with the second parameter of the OS Parameter specification of the DLI, DBB or BMP step. The PSB field is compared with the third parameter of the OS Parameter specification of the DLI, DBB or BMP step. If the PSB parameter is omitted, the Program name is used as the PSB name, much like that of IMS’s processes. In the event that DB2 is access by this step and a program name override is utilized by a DDITV02 DD statement, the OS Parameter will still be used as the Program name for selection criteria purposes. During step execution, the override will be processed normally.

Options

Using options, you can define what action Job Optimizer for IMS should perform for selected batch jobs.

Action

If the ACTION field is left blank, the Default Action as described in the IMS Policy Information section will be in effect for this selection block. Valid parameters for this field are

- SERIALIZE - the SERIALIZE option will, when all selection criteria are ‘a match’ cause the IMS Step to run on the MVS image it was initiated on, and will not allow any other Step, IMS or non-IMS, to run concurrently.

- SPLIT - The SPLIT option will, when all selection criteria are ‘a match’ allow the IMS Step to run on any MVS image within the BatchPlex that has Job Optimizer for IMS running as a partner. This option will also attempt to remove any IMS specific reasons that would prevent this step and other steps, IMS and NON-IMS, from running concurrently.Job Optimizer for IMS.

- DLIREAD - The DLIREAD option provides the capability of removing the restriction of having UPDATE PCB’s in a PSB preventing step concurrency. When this option is specified, and no DLI UPDATE calls have been made using the UPDATE PCB’s during the Job’s prior executions under the Job Optimizer’s history gathering (analysis) phase, Job Optimizer for IMS will remove the UPDATE PCB restriction. If however, a DLI UPDATE occurs during step execution, Job Optimizer for IMS will cause the DLI update to fail and return an “AM” DLI status code in the status code field of the DLI DB PCB mask. It is recommended that this option be used only if the user is sure that DLI Status code checking is present in all jobs, which may match the selection criteria block that specifies DLIREAD. This option is only valid for DLI and DBB steps. When this parameter is encountered by a BMP step, the ACTION is treated as SPLIT.

- DLIABEND - The DLIABEND option provides the capability of removing the restriction of having UPDATE PCB's in a PSB preventing step concurrency. When this option is specified, and no DLI UPDATE calls have been made using the UPDATE PCB's during the Job's prior executions under the Job Optimizer's history gathering (analysis) phase, Job Optimizer for IMS will remove the UPDATE PCB restriction. If however, a DLI UPDATE occurs during step execution, Job Optimizer for IMS will cause the DLI update to fail and cause a User 4070 Abend. It is recommended that this option be used if the user is unsure that DLI Status code checking is present in all jobs and provides an alternative to DLIREAD. This option is only valid for DLI and DBB steps. When this parameter is encountered by a BMP step, the ACTION is treated as SPLIT.

IMS Group

The IMS Group field is only applicable to IMS BMP steps. For DLI and DBB IMS steps, this field is ignored. This field allows the user to group IMS subsystems, which share RECON datasets and databases and which may run on any MVS image within the BatchPlex. This grouping allows Job Optimizer for IMS to route BMP steps which have SPLIT as their ACTION parameter, to other MVS images within the BatchPlex.

The IMS Group field allows a 1 to 8 character name to identify the group.

The RECON dataset field requires the RECON1 dataset name that the group of IMS subsystems uses. This is the dataset name that is specified in the RECON1 MDA member if dynamic allocation is used for the RECON datasets, or the dataset name specified in the RECON1 DD statement.

The Subsystem ID field allows up to 32 IMS subsystems to be associated with this group. When the IMS subsystems are started, their RECON1 dataset is cross-checked with all IMS groups defined within an active IMS Policy and if the RECON1 datasets do not match, the IMS subsystem is NOT included in the group. A single RECON dataset name may be used in more than one IMS group, and IMS subsystem ID's may also be specified in multiple IMS groups. If this option is used, the IMSGROUP indicated is examined and a list of the MVS images in the BatchPlex is built based on where the candidate IMS subsystem ID's are currently running. This list of MVS images is passed to Job Optimizer, who decides which MVS image is the best candidate for the step to execute. The BMP step is then routed to the selected MVS image. The Job Optimizer for IMS executing on the targeted MVS image determines which IMS subsystem (as there may be multiple candidate IMS subsystems) the BMP step will run under. Job Optimizer for IMS then passes the new IMS subsystem name to the IMS BMP region controller to permit execution on the selected IMS subsystem.

Report

The REPORT option is a YES/NO field that specifies if a Job Optimizer for IMS report should be produced for the IMS step. If YES (Y) is specified the report is written to a dynamically allocated SYSOUT DD (@BSBRPT@ is the DD name) and is written to the SYSOUT output class specified by the “MSGCLASS=” parameter on the JOB card.

Creating Policies for Job Optimizer for IMS

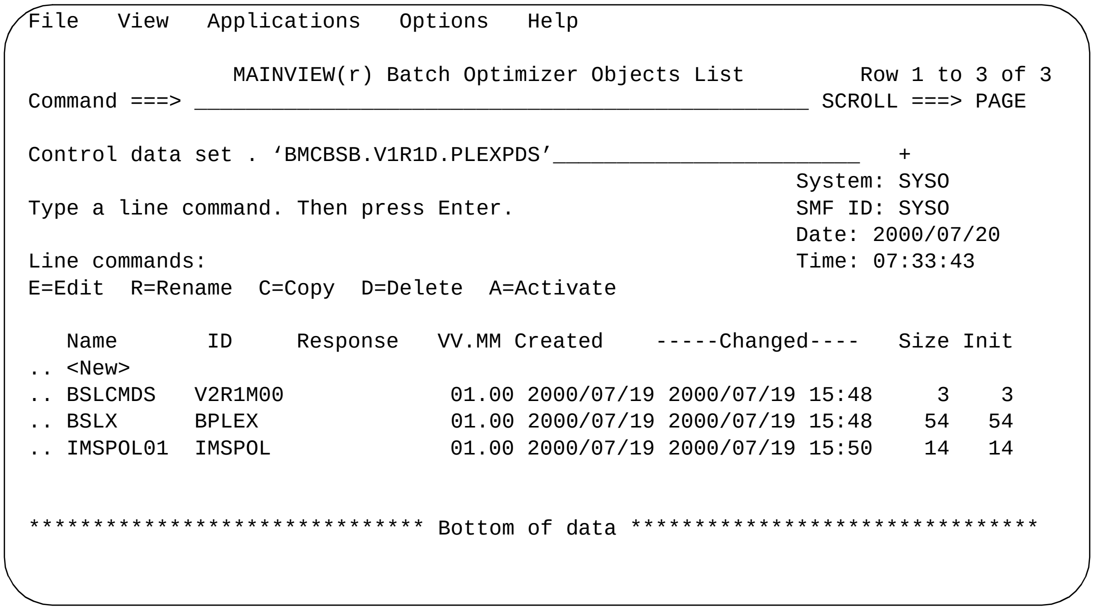
In order for Job Optimizer for IMS to consider taking action on a batch job, the base Job Optimizer product must first select the job for processing. This can be accomplished by ensuring that the current Job Optimizer Job Policy contains a Selection Criteria (SELDEF) entry, which will cause Job Optimizer to take action on the job.

The Job Optimizer’s Selection Definition will cause action to be taken on the job, but in order to provide maximum integrity and flexibility, a separate policy, specifically intended for IMS DLI/DBB batch and BMP steps is required. This IMS Policy, while the primary selection criteria may be the Jobname, has the capability of specifying other criteria, such as Program name and PSB name, to allow the decision to split to be made at the STEP level.

An example of an IMS Policy can be found in the Sample Plex dataset member IMSPOL00. The Sample Plex dataset was downloaded to your system DASD from product installation tape dataset BMC.BSS.SAMPLIB.

IMS Policies may be created, edited and viewed, using the MAINVIEW Batch Optimizer Dialog. The Creating IMS Policy task descriptions in this chapter assume that you are familiar with the dialog’s initial panel sequence and basic navigation techniques.

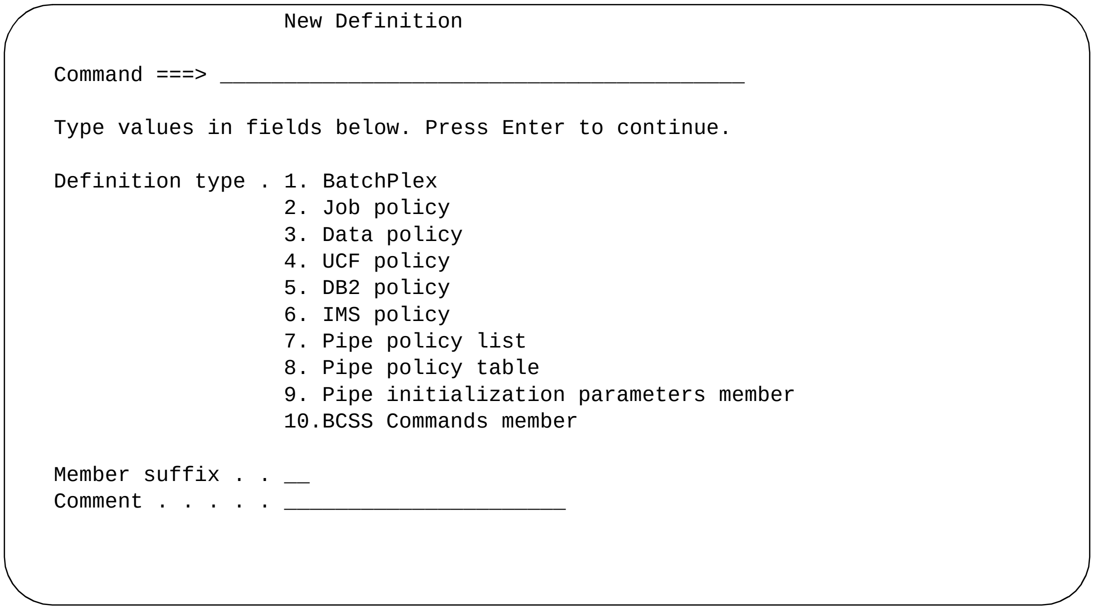
Figure 14-1 MAINVIEW Batch Optimizer Objects List Panel



```
File   View   Applications   Options   Help

                MAINVIEW(r) Batch Optimizer Objects List         Row 1 to 3 of 3
Command ===> ________________________________________________ SCROLL ===> PAGE

Control data set . ‘BMCBSB.V1R1D.PLEXPDS’________________________   +
                                                            System: SYSO
Type a line command. Then press Enter.                      SMF ID: SYSO
                                                            Date: 2000/07/20
Line commands:                                              Time: 07:33:43
E=Edit  R=Rename  C=Copy  D=Delete  A=Activate

   Name       ID     Response   VV.MM Created    -----Changed----   Size Init
.. <New>
.. BSLCMDS   V2R1M00             01.00 2000/07/19 2000/07/19 15:48     3    3
.. BSLX      BPLEX               01.00 2000/07/19 2000/07/19 15:48    54   54
.. IMSPOL01  IMSPOL              01.00 2000/07/19 2000/07/19 15:50    14   14


******************************* Bottom of data ********************************
```

To create a new IMS policy, enter the Edit line command for the “<NEW>” name entry of the MAINVIEW Batch Optimizer Objects List panel.

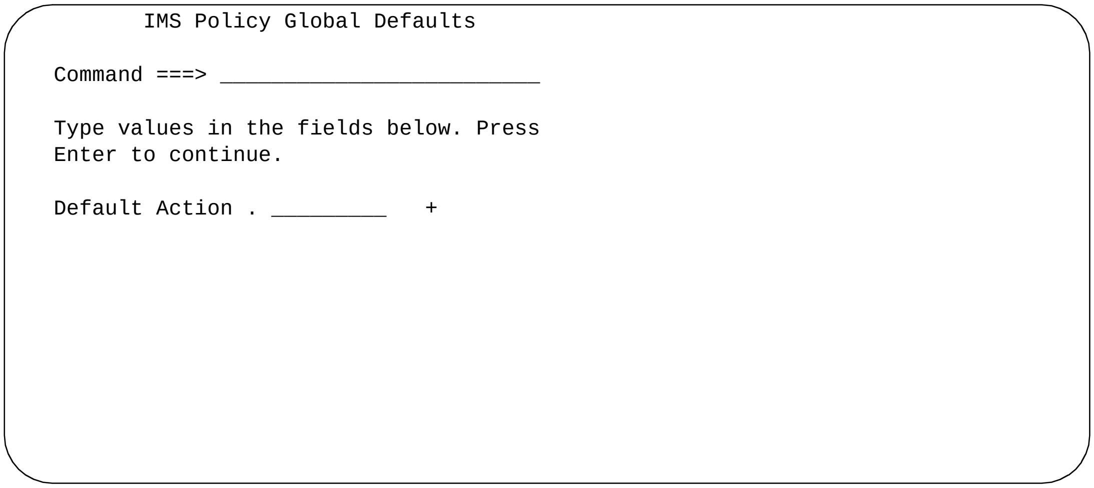
Figure 14-2 MAINVIEW Batch Optimizer New Definition Panel



```
                  New Definition

 Command ===> _________________________________________

 Type values in fields below. Press Enter to continue.

 Definition type . 1. BatchPlex
                   2. Job policy
                   3. Data policy
                   4. UCF policy
                   5. DB2 policy
                   6. IMS policy
                   7. Pipe policy list
                   8. Pipe policy table
                   9. Pipe initialization parameters member
                   10.BCSS Commands member

 Member suffix . . __
 Comment . . . . . ______________________
```

You will then be prompted to provide a Definition Type for the new object. Select **6** to create an IMS policy. The prefix IMSPOL will be used and a two-character member suffix is also required to complete the name. A 22-byte comment field is also available for use.

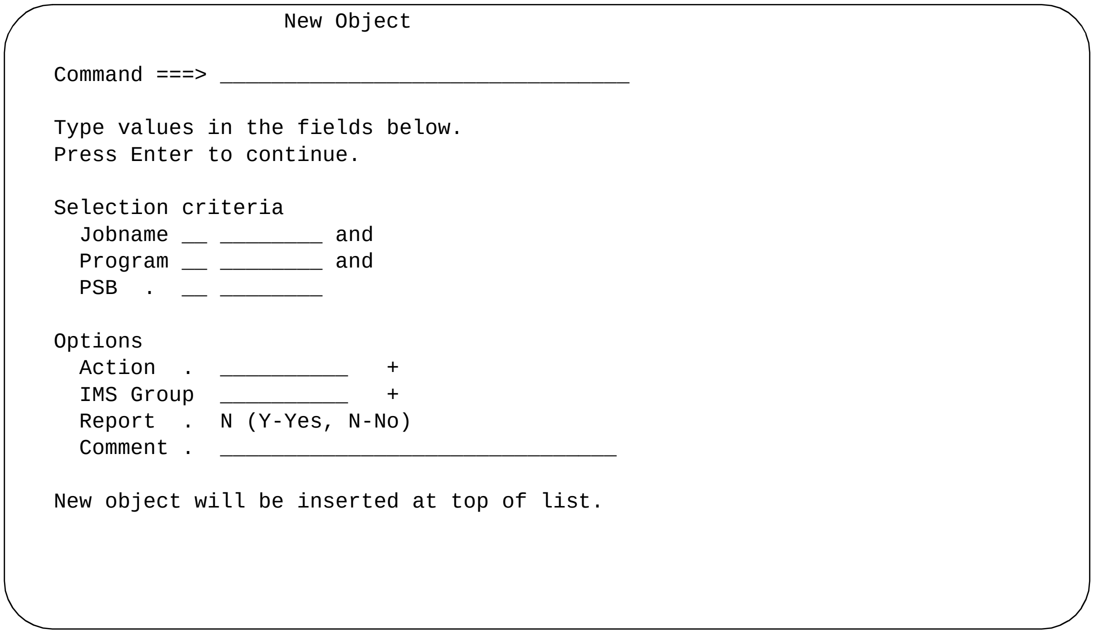
Figure 14-3 IMS Policy Global Defaults Panel



```
       IMS Policy Global Defaults

Command ===> _________________________

Type values in the fields below. Press
Enter to continue.

Default Action . _________   +
```

The DEFAULT ACTION is currently the only GLOBAL parameter required. This parameter supplies the ACTION that Job Optimizer for IMS is to take when an IMS step has been selected for processing, but no step specific action was coded in the selection block. Click on, or position your cursor on the ‘plus’ (+) sign and hit enter, to see a list of valid ACTION options displayed. The “SERIALIZE” action is recommended for the Global Default. See “Options” on page 14-10 for a list of valid Action options, their descriptions and impact.

Figure 14-4 New Object Panel



```
                  New Object

Command ===> ________________________________

Type values in the fields below.
Press Enter to continue.

Selection criteria
  Jobname __ ________ and
  Program __ ________ and
  PSB  .  __ ________

Options
  Action  .  __________   +
  IMS Group  __________   +
  Report  .  N (Y-Yes, N-No)
  Comment .  _______________________________

New object will be inserted at top of list.
```

There are two sections to this panel, Selection criteria and Options.

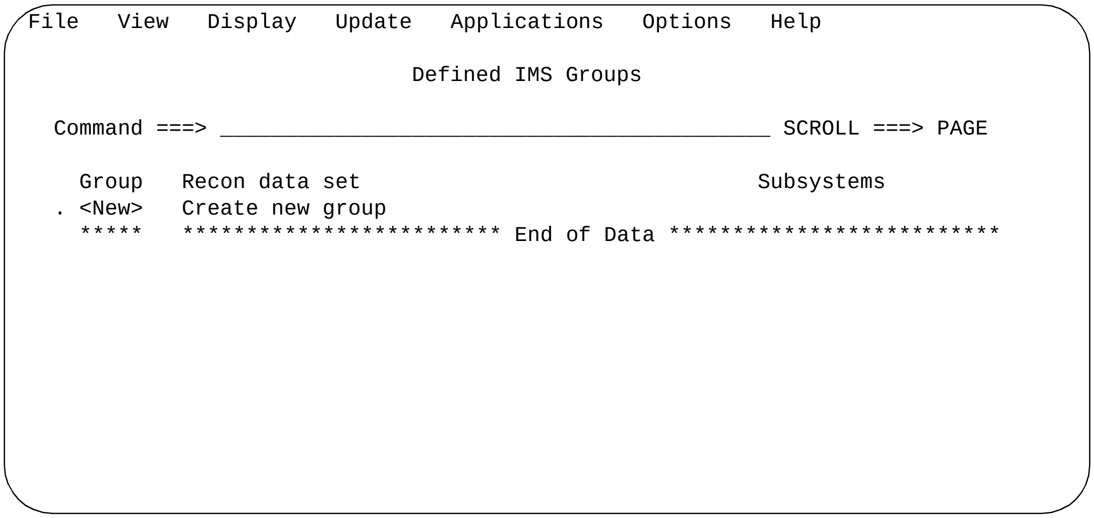
The Selection criteria section defines which jobs and steps within the job, Job Optimizer for IMS should consider processing. See “Options” on page 14-10 for details on comparison operands and valid Jobname, Program name and PSB name field contents and their impact.

The Options section describes what actions should be taken when the Selection criterion is met.

The ACTION parameter determines the process that Job Optimizer for IMS is to take when an IMS step has been selected for processing. If this field is left blank, the Global Policy Default Action will be in effect. Click on, or position your cursor on the ‘plus’ (+) sign and hit enter, to have a list of valid ACTION options displayed. See “Options” on page 14-10 for a list of valid Action options, their descriptions and impact.

The IMSGROUP is an optional field that enables Job Optimizer for IMS to dynamically route BMP steps, up to a possible 32 IMS subsystems which share a common set of RECON datasets, potentially on multiple MVS images without any JCL changes. Click on, or position your cursor on the ‘plus’ (+) sign and hit enter, to have a list of valid Group names.

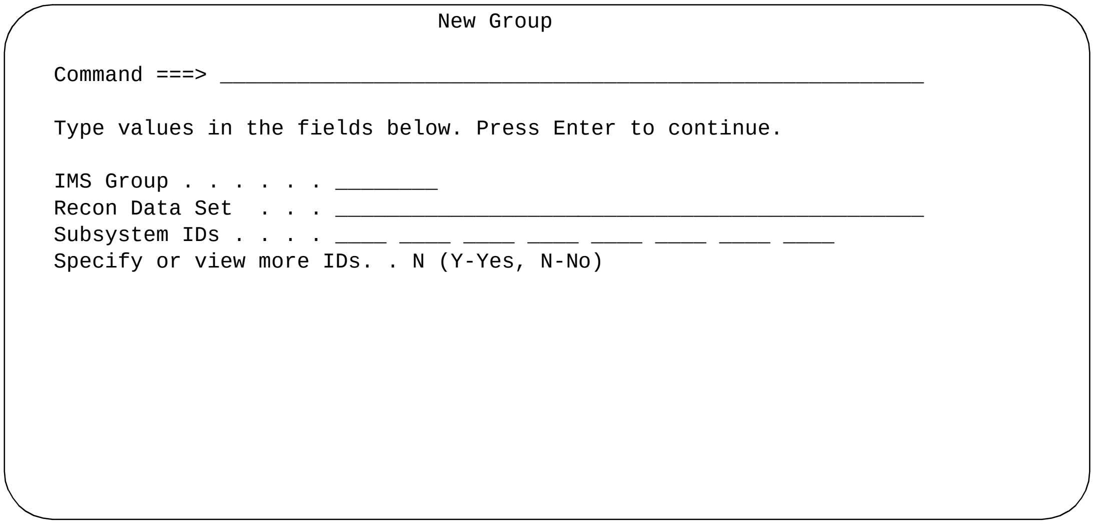
Figure 14-5 Defined IMS Groups Panel



```
File   View   Display   Update   Applications   Options   Help

                            Defined IMS Groups

 Command ===> ___________________________________________ SCROLL ===> PAGE

   Group   Recon data set                              Subsystems
 . <New>   Create new group
   *****   ************************* End of Data **************************
```

To create a new Group name, enter the Edit line command for the “<NEW>“name entry of the Defined IMS Groups panel.

Figure 14-6 New Group Panel

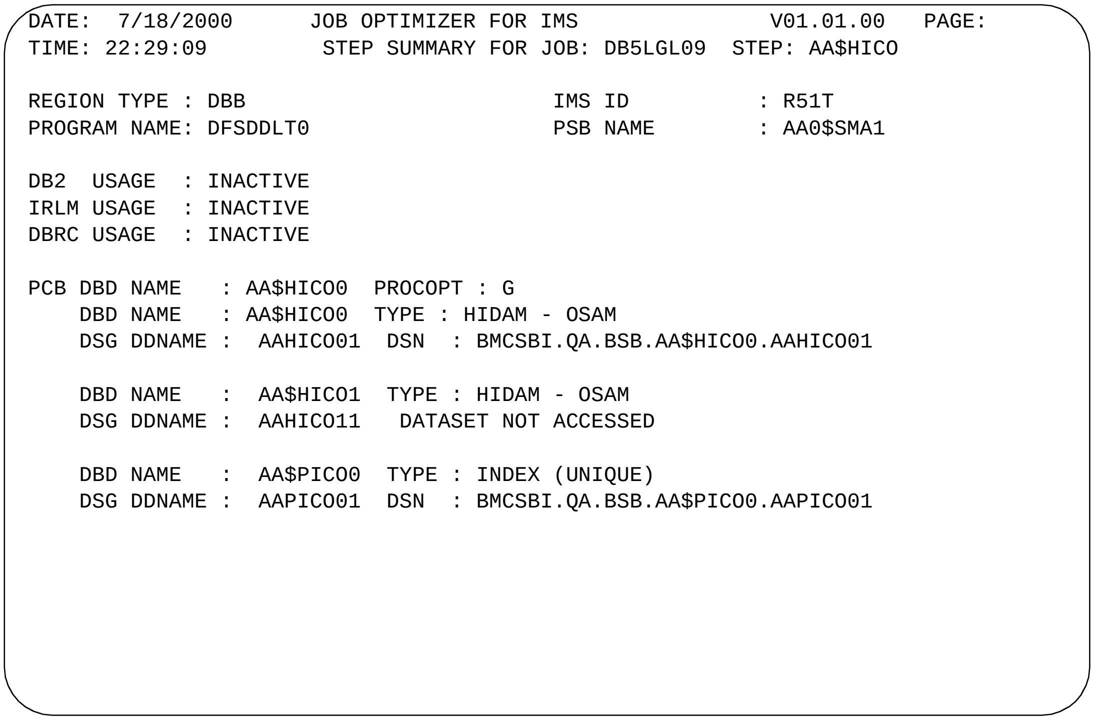


```
                              New Group

Command ===> _______________________________________________________

Type values in the fields below. Press Enter to continue.

IMS Group . . . . . . ________
Recon Data Set  . . . ______________________________________________
Subsystem IDs . . . . ____ ____ ____ ____ ____ ____ ____ ____
Specify or view more IDs. . N (Y-Yes, N-No)
```

On the New Group panel, type the name of the group you wish to create in the IMS Group field. For the RECON dataset, use the RECON1 dataset name of the RECON dataset group, which will be shared by the IMS subsystem ID's that are to be entered. You may enter up to 8 IMS subsystem id's on the New Group primary panel. If you wish to specify more, up to 32 may be entered on a secondary panel when a **Y** is placed in the 'Specify or view more Ids' field. When complete hit "ENTER" and you will be returned to the New Object Menu.

The last option on the New Object Panel, is the REPORT option, which is a YES/NO field that specifies if a Job Optimizer for IMS report should be produced for the IMS step. If YES (Y) is specified the report is written to a dynamically allocated SYSOUT DD (@BSBRPT@ is the DD name) and is written to the SYSOUT output class specified by the "MSGCLASS=" parameter on the JOB card. An example of the report is shown below.

Figure 14-7 Job Optimizer for IMS Report Example

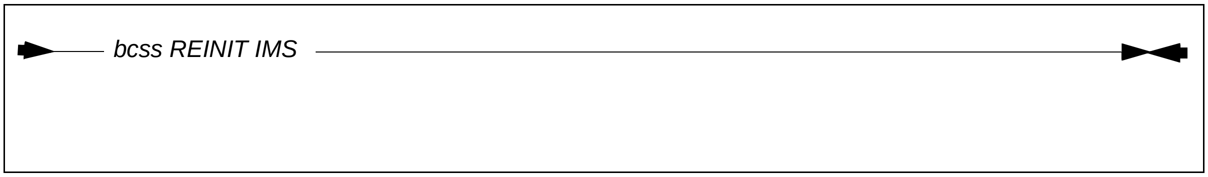


```
DATE:  7/18/2000      JOB OPTIMIZER FOR IMS               V01.01.00   PAGE:
TIME: 22:29:09         STEP SUMMARY FOR JOB: DB5LGL09  STEP: AA$HICO

REGION TYPE : DBB                       IMS ID          : R51T
PROGRAM NAME: DFSDDLT0                  PSB NAME        : AA0$SMA1

DB2  USAGE  : INACTIVE
IRLM USAGE  : INACTIVE
DBRC USAGE  : INACTIVE

PCB DBD NAME   : AA$HICO0  PROCOPT : G
    DBD NAME   : AA$HICO0  TYPE : HIDAM - OSAM
    DSG DDNAME :  AAHICO01  DSN  : BMCSBI.QA.BSB.AA$HICO0.AAHICO01

    DBD NAME   :  AA$HICO1  TYPE : HIDAM - OSAM
    DSG DDNAME :  AAHICO11   DATASET NOT ACCESSED

    DBD NAME   :  AA$PICO0  TYPE : INDEX (UNIQUE)
    DSG DDNAME :  AAPICO01  DSN  : BMCSBI.QA.BSB.AA$PICO0.AAPICO01
```

Activating Job Optimizer for IMS Processing

To activate Job Optimizer for IMS processing, issue the following command from the MVS console:

Figure 14-8 Activating Job Optimizer for IMS Processing



```
bcss REINIT IMS
```

bcss is the ID of the MAINVIEW Batch Optimizer subsystem that is active on the MVS image.

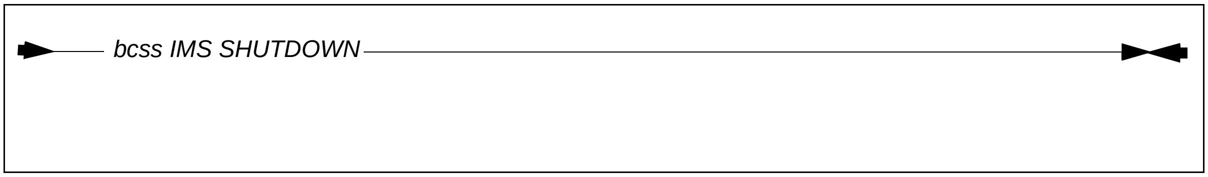
Successful initialization will result in a BMC194001I message being issued.

Note: As an alternative to the above step, you can activate Job Optimizer for IMS processing by adding the REINIT IMS command to the BatchPlex Global MAINVIEW Batch Optimizer Subsystems Options Commands member. If specified, you can also add the MVS Image MAINVIEW Batch Optimizer Subsystem Options Commands member.

Terminating Job Optimizer for IMS

To terminate Job Optimizer for IMS processing, issue the following command from the MVS console:

Figure 14-9 Terminating Job Optimizer for IMS Processing



```
bcss IMS SHUTDOWN
```

bcss is the ID of the MAINVIEW Batch Optimizer subsystem that is active on the MVS image.

Successful terminations will result in a BMC194002I message being issued.

Activating an IMS Policy

To activate an IMS policy, issue the following command from the MVS console:

Figure 14-10 Activating an IMS Policy

```
bcss IMS POLICY ACTIVATE policyname
```

bcss is the ID of the MAINVIEW Batch Optimizer subsystem that is active on the MVS image.

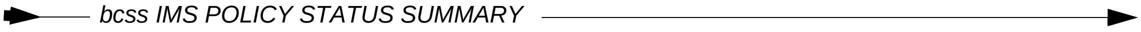
Note: As an alternative to the above step, from the User Interface's MAINVIEW Batch Optimizer Objects List panel, (panel BSSP00IP), type **A** (for Activate) in the command column on the line which contains the name of the IMS policy you wish to activate.

Successful activation will result in a BMC194027I message being issued. If an IMS policy was already active when the command was issued, a BMC194026I message indicating the name of the IMS policy that was removed.

Displaying the Active IMS Policy

To display the name of the active IMS policy, issue the following command from the MVS console:

Figure 14-11 Displaying the Active IMS Policy



bcss IMS POLICY STATUS SUMMARY

bcss is the ID of the MAINVIEW Batch Optimizer subsystem that is active on the MVS image.

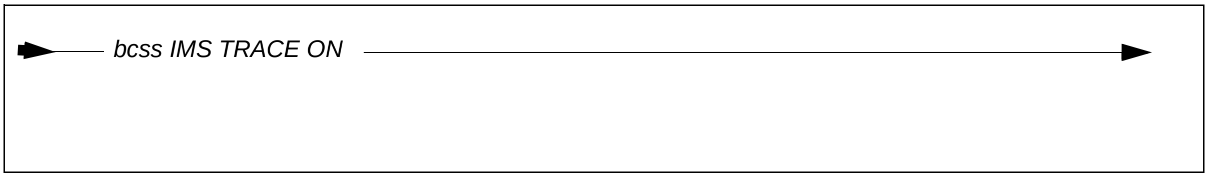
Message BMC194035I will be issued indicating the currently active IMS POLICY name. If there was not an active IMS POLICY, a BMC194035I message will be issued.

Activating Tracing for Job Optimizer for IMS Modules

Tracing of Job Optimizer for IMS modules should be done *only* when requested by BMC Software support staff. Tracing for the base Job Optimizer component *must* be ACTIVE.

To start tracing Job Optimizer for IMS modules, issue the following command from the MVS console:

Figure 14-12 Activating Tracing for IMS Modules



```
bcss IMS TRACE ON
```

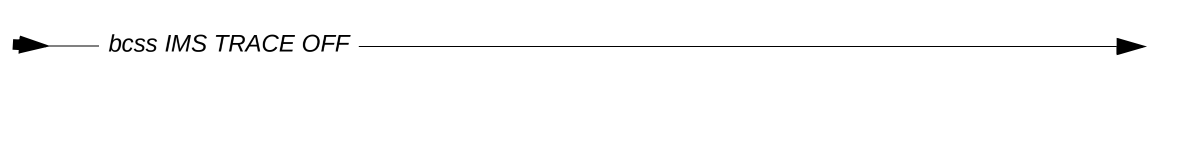
bcss is the ID of the MAINVIEW Batch Optimizer subsystem that is active on the MVS image.

Successful activation will result in a BMC194028I message being issued indicating that tracing is on.

Terminating Tracing for Job Optimizer for IMS Modules

To terminate tracing of Job Optimizer for IMS modules, issue the following command from the MVS console:

Figure 14-13 Terminating Tracing for IMS Modules



```
►►── bcss IMS TRACE OFF ──────────────────────────────────────────►
```

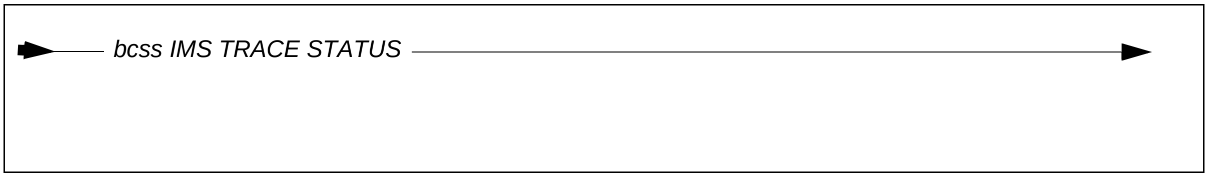
bcss is the ID of the MAINVIEW Batch Optimizer subsystem that is active on the MVS image.

Successful activation will result in a BMC194028I message being issued indicating that tracing is off.

Displaying Status of Job Optimizer for IMS Tracing

To display the status of Job Optimizer for IMS tracing, issue the following command from the MVS console:

Figure 14-14 Displaying the Status of IMS Tracing



```
bcss IMS TRACE STATUS
```

bcss is the ID of the MAINVIEW Batch Optimizer subsystem that is active on the MVS image.

This command will result in a BMC194028I message being issued indicating that tracing is on or off.

Appendix A Job Optimizer Reports

This appendix describes the reports that are generated by Job Optimizer. This appendix discusses the following topics:

BMC Software, Inc., Confidential and Proprietary Information

How to Obtain Reports about Job Performance

You can obtain information about the potential or actual benefits of Job Optimizer for the jobs in your site. Whether the report presents potential or actual elapsed times depends on the source of the data.

The source data for these reports can be data from SMF or the history data set. When you use SMF data, the report presents potential elapsed times. When you use the history data set, the report presents either potential elapsed times for jobs in which MAINVIEW Batch Optimizer is performing job analysis or actual elapsed times from jobs in which MAINVIEW Batch Optimizer has already split the job.

You can generate two types of reports:

- The Summary Information report presents a single line of information about each job. You can use this report to determine which jobs have longer run times and higher predicted savings. Remember that actual performance results depend on your workload and environment. Figure Figure A-1 on page A-3 shows a sample of this report.

- The Detail Information report presents a graphical view of the job statistics. Information about processing and elapsed times is provided along with one of several graphical lines, which represent job performance before and after using Job Optimizer job processing. You can use this report to view graphically how the jobs might be split if it is intercepted by Job Optimizer for job performance processing or how jobs are being split currently. Figure A-2 on page A-5 shows a sample of this report.

Summary Information Report

Figure A-1 shows the Summary Information report.

Figure A-1 Candidate Report - Summary Information

```
***************************************************************************
******************
* JOBID: BSLBJEXT(JOB01088)  JOBSTEP: REPORT   DDNAME: REPORT   DATE:
00/07/25  TIME: 10:51:32     *
***************************************************************************
******************
                            Job Optimizer Performance Report
                      PAGE: 0001
                                Summary Information
                    Normal   Optimized Predicted Percent            History
Jobname  Jobid      Time     Time      Savings   Savings  MVSImage Source
__(5)___ ________  ___(2)____ __(4)___ ___(1)___ __(3)___ __(6)___ ________
ACCT03   JOB08161     0:00:39  0:00:23   0:00:16       41 SYSQ
OPTIMIZED
ACCT05   JOB08164     0:00:38  0:00:22   0:00:16       42 SYSQ
OPTIMIZED
ACCT     JOB05848     0:00:26  0:00:12   0:00:14       54 SYSJ     ANALYZED
ACCT06   JOB08167     0:00:36  0:00:23   0:00:13       37 SYSJ
OPTIMIZED
ACCT02   JOB08158     0:00:36  0:00:24   0:00:12       34 SYSJ
OPTIMIZED
ACCT07   JOB08169     0:00:32  0:00:19   0:00:12       39 SYSQ
OPTIMIZED
ACCT04   JOB08163     0:00:34  0:00:23   0:00:11       32 SYSJ
OPTIMIZED
ACCT08   JOB08171     0:00:30  0:00:19   0:00:11       36 SYSQ
OPTIMIZED
ACCT11   JOB08187     0:00:21  0:00:14   0:00:06       31 SYSQ
OPTIMIZED
ACCT09   JOB08175     0:00:20  0:00:14   0:00:06       29 SYSJ
OPTIMIZED
ACCT10   JOB08180     0:00:19  0:00:14   0:00:05       26 SYSQ
OPTIMIZED
ACCT12   JOB08190     0:00:18  0:00:14   0:00:04       22 SYSQ
OPTIMIZED
ACCT01   JOB08157     0:00:26  0:00:23   0:00:03       13 SYSJ
OPTIMIZED
 Number of jobs processed. . . . . . . 13

******************************** END OF DATA
************************************************
```

Table A-1 describes the fields in the Summary Information report.

Table A-1 Candidate Report - Field Descriptions

Field	Description
Jobname	Name of the job for which the statistics were collected.
Jobid	Job identification number of the job for which the statistics were collected.
Normal Time	Total time required by all of the steps in the job to complete execution without MAINVIEW Batch Optimizer job performance processing. The format of this value is hh:mm:ss.
Elapsed Time	Depending on the source of the data (SMF or history data set), elapsed time is the predicted or actual time that all of the steps in the job would require to complete execution with MAINVIEW Batch Optimizer job performance processing. The format of this value is hh:mm:ss.
Predicted Savings	Difference between the application time and the elapsed time. The format of this value is hh:mm:ss.
Percent Reduced	Predicted savings as a percent of the application time. This value is calculated from the actual SMF data or snapshot from History file, which is written in hundredths of a second.
MVS Image	System where the job started and ran in the JES initiator.
History Source	**This field contains the source of the data and can be one of these values:** • **Analyze** - Source is the history file, but the job was not optimized. Savings are predicted. • **Optimized** - MAINVIEW Batch Optimizer was invoked and the job obtains performance enhancements. The source was the MAINVIEW Batch Optimizer history file. • SMF - Source was the SMF records.
Number of Jobs Processed	Number of jobs included in the report.
Number of segments in error	Number of jobs in which errors were detected while reading the input data. This field is not normally displayed.

Note: The format of all fields that show time values is *hh:mm:ss*.

Detail Information Report

Figure A-2 shows the Detail Information report.

Figure A-2 Candidate Report - Detail Information

```
***************************************************************************
*************************
*  JOBID: BSLBJEXT(JOB01094)   JOBSTEP: REPORT  DDNAME: REPORT  DATE:
00/07/25    TIME: 10:58:41   *
***************************************************************************
*************************
                      Job Optimizer Performance Report
PAGE: 0001
                                   Detail Information

       ACCT01   JOB08157
                History source:                  Optimized
                Elapsed time (Normal):            0:00:26_
                Step executor time                0:00:18_
                Elapsed time (Optimized):         0:00:23_
                Savings (Actual):                 0:00:03   13%

Step |  Execution (Optimized)
   1 |1111111111111
   2 |     2222222
   3 |     333333333
   4 |                 44444444444444444444444444444
   5 |                   55555555555555555
   6 |                  66666666666
   7 |                                            77777777777
   8 |                                                         88

|---------|---------|---------|---------|---------|---------|---------|----
-----|---------|--------|
     0         10        20        30        40        50        60
70        80        90            Number of jobs processed. . . . . . . 1

***************************************** END OF DATA
**********************************************
```

Table A-2 describes the fields in the Summary Information report.

Table A-2 Detail Information Report - Field Descriptions

Field	Description
History source	This field contains the source of the data and can be one of three values: • Analyze—the history data set when the job has been analyzed • Split—the history data set when the job has been split • SMF—SMF records
Normal elapsed time	Cumulative length of time that all of the steps in the job required to complete execution without optimization.
Step executor time	Number of hh:mm:ss the job ran in the JES initiator. Does not include split steps.
Step executor time	Amount of system time required to perform the job switching tasks. This time is usually less than 1 second.
Elapsed time optimized	Actual clock time that the job required to complete execution.
Predicted Savings	Difference between the application time and the elapsed time.
Step (1, 2, and so on)	Graphical representation of the percentage of time spent by each job step and the starting and ending times relative to the other steps. If the source of the report is the history data set and MAINVIEW Batch Optimizer is analyzing the job, or, if the source of the report is SMF data, the report presents two graphical lines. The first line (Prior to MAINVIEW Batch Optimizer) shows how the job runs without MAINVIEW Batch Optimizer job performance processing. The second line (Post MAINVIEW Batch Optimizer) predicts how the job will run with MAINVIEW Batch Optimizer job performance processing. If the source of the report is the history data set and MAINVIEW Batch Optimizer has already split the job, the report presents one graphical line. Again, actual performance results depend on your workload and your environment.
Number of jobs processed	Number of jobs included in the report.

Note: The format of all fields that show time values is *hh:mm:ss*.

Using SMF Data Sets as the Report Source

To use Systems Management Facilities (SMF) data sets as the source for generating the Summary Information report or the Detail Information report, use the JCL in member BSLBJRPT in data set SAMPLIB.

The sample JCL is divided into the following steps:

- Step 1 deletes allocated data sets from previous executions of this utility.
- Step 2 uses the Populate History utility to extract data from SMF data sets and create a select file and a discard file.

  The select file contains the names of jobs that match your selection criteria. The select file is used as input to the third step. The discard file, which is optional, contains the job names that did not match your selection criteria.
- Step 3 generates the Summary Information report or the Detail Information report from the select file.

To generate either report, modify the JCL as described in Table A-3 and submit it. The steps should complete with a condition code of zero (0).

Step 1 If you did not install the MAINVIEW Batch Optimizer load modules in the MVS LNKLST, uncomment the STEPLIB and EXITLIB DD statements and verify that the data set names are correct.

Step 1 Verify that the data set name provided on the SORTLIB DD statement is correct.

Step 2 Include the name of your SMF data sets on the SORTIN DD statement.

Step 3 Concatenate as many SMF data sets as you like to collect as much data as you want.

Step 4 Check the allocation sizes of the data sets that are referenced by the SELECT and DISCARD DD statements.

Step 5 Increase or decrease the allocation sizes based on the amount of SMF data that you are using as input.

Step 6 Replace all instances of *"?SHAREDQUAL?"* with a valid high-level qualifier.

Step 7 Replace all instances of *"?DISKUNIT?"* with a valid unit name that identifies disk devices that are used for permanent storage.

Step 8 Replace all instances of *"?WORKUNIT?"* with a valid unit name that identifies disk devices that are used for temporary storage.

Step 9 Uncomment one of the PARM parameters on the REPORT EXEC statement. Use S for the Summary Information report and D for the Detail Information report. If you do not uncomment either PARM parameter, the default is to generate the Summary Information report.

Step 10 The LINES parameter on the RPT1CNTL DD statement controls page eject. The default is 60 lines per page. The maximum value is 99 lines per page.

Step 11 Review and modify the parameters that are listed after the SMFCNTL DD statement as needed. All parameters must start in column 1.

Note: The DATE, START, END, and SID parameters are the same as those used with the SMF Dump program (IFASMFDP).

Table A-3 Summary Information Report and Detail Information - Field Descriptions

Field	Description
DATE(*yyyyddd,yyyyddd*)	Select records that were recorded within the specified range of dates (*yyyy* is the year and *ddd* is the Julian date). The value for *ddd* cannot exceed 366. If DATE is specified, you must include both a start date and an end date. If DATE is not specified, the default is DATE(1900000,2099366).
START(*hhmm*)	Select records that were recorded after the START time but before the END time (*hh* is hours and *mm* is minutes based on a 24-hour clock). If START is not specified, the default is START(0000).
END(*hhmm*)	Select records that are identified with times less than, or equal to, *hhmm* (hh is hours and *mm* is minutes based on a 24-hour clock). If END is not specified, the default is END(2400).
JOBNAME(*jobname, jobname*,...)	Select records that pertain to jobs with specific names or generic name patterns (*jobname* is a specific job name or the generic job name pattern that contains one or more asterisks (*)). You can specify up to 1,000 specifications. To continue your specifications to another line, end the current line with a comma. If JOBNAME is not specified, the default is JOBNAME(*).
SELECT(YES)	Generate the select file and, optionally, the discard file. The discard file is generated only if you include the DISCARD DD statement. If SELECT(YES) is not specified, no records are written to the select file.
MINEXECTIME(*mmss*)	Do not select jobs whose elapsed time is less than the specified amount (*mm* is minutes and *ss* is seconds). If MINEXECTIME is not specified, the default is MINEXECTIME(0).

Populating the History Data Set

You can use your existing SMF data as a starting point for choosing candidate jobs for Job Optimizer's job performance processing. Job Optimizer uses the job structure and data access information that it collects during job analysis to optimize job performance processing. Rather than waiting for jobs to accumulate sufficient historical information for job optimization, you can populate the history data set with SMF data from previously run jobs so that Job Optimizer can begin optimizing jobs immediately.

To populate the history data set with SMF data, your installation must collect the following SMF record types:

- 14
- 15
- 17
- 18
- 30 (subtypes 1, 4, and 5),
- 61
- 63
- 64
- 65
- 66
- 67
- 68
- 101

To be sure, check your active SMFPRM*xx* member of PARMLIB.

It is a good idea to populate the history data set in the following circumstances:

- When the history data set is initially created during customization. You can seed this empty data set with information about previous job runs.

- If you decide later to use job performance processing for a new group of jobs, you can populate the history data set with additional SMF data.

- If you are migrating history records from one history data set to another (such as moving from test to production).

You can use either of the following methods to populate the history data set:

- Populate History utility–This utility provides a record selection and filtering facility that selects SMF records based on system ID, time, date, or job name (the facility is patterned after the facility provided by the utility program IFASMFDP).

  If you do not specify selection criteria, the utility prepares history data set records for all jobs that are referenced by the input SMF data set. This method populates the history data set with data for jobs that might or might not be good job performance candidates.

- Load utility–This utility allows you to be more selective about the jobs that you use to populate the history data set.

  When you use SMF data as the generating source for a job performance report, a select file is produced with the report. You can use the report to determine which jobs would make good job performance candidates. Then, you can use the select file as input to the Load utility, which extracts information about selected jobs and uses only the selected information to populate the history data set.

Running the Populate History Utility

The Populate History utility sorts the selected SMF records by system ID, job name, and job start date and start time. The utility requires access to your system's SORT utility. The utility generates a summary report of the numbers and types of records and jobs that were selected for inclusion in the history data set.

The MAINVIEW Batch Optimizer subsystem must be running when you use the Populate History utility.

To run the Populate History utility, modify the JCL in member BSLBJRUN of data set SAMPLIB as described and submit the JCL. The steps will complete with a condition code of zero (0).

- If you did not install the MAINVIEW Batch Optimizer load modules in your MVS LNKLST, uncomment the STEPLIB and EXITLIB DD statements and verify that the data set names are correct.

- Verify that the data set name provided on the SORTLIB DD statement is correct.

- On the SORTIN DD statement, include the name of the data set that contains the SMF-formatted records.
- Review and modify as needed the following parameters, which are listed after the SMFCNTL DD statement (all parameters must start in column 1):

Note: The DATE, START, END, and SID parameters are the same as those used with the SMF Dump program (IFASMFDP).

Table A-4 Populate History Utility Summary Report Parameters

Parameter	Description
SUBSYSNAME(*ssid*)	Names the MAINVIEW Batch Optimizer subsystem whose history data set you want to populate (*ssid* is a MAINVIEW Batch Optimizer subsystem ID). The subsystem procedure identifies the name of the history data set. The default subsystem ID is bcss (as defined during the customization process).
DATE(*yyyyddd,yyyyddd*)	Select records that were recorded within the specified range of dates (*yyyy* is the year and *ddd* is the Julian date). The value for *ddd* cannot exceed 366. If DATE is specified, you must include both a start date and an end date. If DATE is not specified, the default is DATE(1900000,2099366).
START(*hhmm*)	Select records that were recorded after the START time but before the END time (*hh* is hours and *mm* is minutes based on a 24-hour clock). If START is not specified, the default is START(0000).
END(*hhmm*)	Select records that are identified with times less than, or equal to, *hhmm* (*hh* is hours and *mm* is minutes based on a 24-hour clock). If END is not specified, the default is END(2400).
SID(*smfname*)	Select records for jobs that run on a specific MVS system (*smfname* is the SMF ID). Specify SID(*) to select records for jobs on any system. You can also use an asterisk in a generic specification. For example, specify SID(SYS*) to select records for jobs that run on any system with an SMF ID that starts with SYS. Use a single asterisk only. Do not imbed the asterisk. For example, SID(S*SA*) is not valid.
JOBNAME(*jobname,jobname,jobname*,...)	Select records that pertain to jobs with specific names or generic name patterns (*jobname* is a specific job name or the generic job name pattern that contains one or more asterisks (*)). You can specify up to 1,000 specifications. To continue your specifications to another line, end the current line with a comma. You can also specify multiple JOBNAME parameters. If JOBNAME is not specified, the default is JOBNAME(*).
MINEXECTIME(*mmss*)	Do not select jobs whose elapsed time is less than the specified amount (*mm* is minutes and *ss* is seconds). If MINEXECTIME is not specified, the default is MINEXECTIME(0).
MAXHISTPCT(*xx*)	Specify how full the history data set should become with SMF records (*xx* is a number from 01 through 75). If MAXHISTPCT is not specified, the default is MAXHISTPCT(50).

BMC Software, Inc., Confidential and Proprietary Information

As Figure A-3 on page A-13 shows, the summary report lists the parameters and values that were used to select records for inclusion in the history data set. The summary report also summarizes the SMF records that the utility used as input and those it selected as output.

If you entered invalid parameters, the summary report contains messages that describe the error.

Figure A-3 Summary Report from the Populate History Utility

```
***************************************************************************
******************
* JOBID: BSLBJEXT(JOB01088)  JOBSTEP: REPORT   DDNAME: REPORT   DATE:
00/07/25  TIME: 10:51:32     *
***************************************************************************
******************
                            Job Optimizer Performance Report
                      PAGE: 0001
                                Summary Information
                    Normal   Optimized Predicted Percent            History
Jobname  Jobid      Time     Time      Savings   Savings  MVSImage Source
__(5)___ _________  ___(2)____ __(4)___ ___(1)___ __(3)___ __(6)___ _________
ACCT03   JOB08161     0:00:39  0:00:23   0:00:16       41 SYSQ
OPTIMIZED
ACCT05   JOB08164     0:00:38  0:00:22   0:00:16       42 SYSQ
OPTIMIZED
ACCT     JOB05848     0:00:26  0:00:12   0:00:14       54 SYSJ     ANALYZED
ACCT06   JOB08167     0:00:36  0:00:23   0:00:13       37 SYSJ
OPTIMIZED
ACCT02   JOB08158     0:00:36  0:00:24   0:00:12       34 SYSJ
OPTIMIZED
ACCT07   JOB08169     0:00:32  0:00:19   0:00:12       39 SYSQ
OPTIMIZED
ACCT04   JOB08163     0:00:34  0:00:23   0:00:11       32 SYSJ
OPTIMIZED
ACCT08   JOB08171     0:00:30  0:00:19   0:00:11       36 SYSQ
OPTIMIZED
ACCT11   JOB08187     0:00:21  0:00:14   0:00:06       31 SYSQ
OPTIMIZED
ACCT09   JOB08175     0:00:20  0:00:14   0:00:06       29 SYSJ
OPTIMIZED
ACCT10   JOB08180     0:00:19  0:00:14   0:00:05       26 SYSQ
OPTIMIZED
ACCT12   JOB08190     0:00:18  0:00:14   0:00:04       22 SYSQ
OPTIMIZED
ACCT01   JOB08157     0:00:26  0:00:23   0:00:03       13 SYSJ
OPTIMIZED
 Number of jobs processed. . . . . . . 13

******************************** END OF DATA
************************************************
```

Input Summary

This section of the report provides totals of the number of SMF records by record type that were examined for possible inclusion in the history data set and the number of SMF records that were included in the history data set.

The following fields are included in Input Summary section of the report:

Table A-5 Populate History Utility Report Input Summary Fields

Field	Description
SMF RECORD TYPE	List of all SMF record types that are extracted by Job Optimizer for examination and possible inclusion in the history data set.
RECORDS INPUT	Number of SMF records in each SMF record type that were examined by Job Optimizer.
RECORDS USED	Number of SMF records in each SMF record type that met the selection criteria and were included in the history data set.

Output Summary

This section of the Populate History Utility report provides a summary of the output records, which are described as follows:

- NEW JOB INSTANCES ADDED TO HISTORY DATA SET FOR SUBSYSTEM *ssid*

  Number of jobs that were written to the history data set that is associated with the identified subsystem (*ssid*). These jobs did not previously appear in the history data set.

- JOB INSTANCES MERGED WITH HISTORY

  Number of jobs that match an existing job name in the history data set. The new SMF data is merged with the existing SMF or Job Optimizer data because the job structure is the same.

- JOB INSTANCES REPLACED IN HISTORY

  Number of jobs that match an existing job in the history data set. The new SMF data replaces the existing SMF data because the job structure has changed. Actual Job Optimizer data is not replaced.

- JOB INSTANCES NOT REPLACED IN HISTORY BECAUSE ACTUAL(NON-SMF) DATA EXISTS

  Number of jobs that match an existing job name in the history data set. The new SMF data is discarded because SMF data will not replace actual Job Optimizer data.

- HISTORY POPULATE NOT COMPLETED BY ACCESS METHOD

  Number of jobs for which errors occurred and, as a result, were not added to the history data set

- JOB INSTANCES WRITTEN TO SELECT FILE

  Number of jobs that were written to the select file.

- RECORDS DISCARDED BECAUSE OF JOB MISMATCH

  Number of SMF records that were not used because the job start record was outside the selected date and time range.

- ONE STEP JOBS DISCARDED

  Number of jobs that had only one step. One-step jobs are not candidates for Job Optimizer.

- INCOMPLETE JOBS DISCARDED

  Number of SMF records that were not used because the job end record was outside the selected date and time range.

Running the Load Utility

The Load utility extracts and sorts selected records from a previously generated report file. The utility requires access to your system’s SORT utility. The utility can sort with either DFSORT or SYNCSORT. The extracted records are used to populate the history data set. The utility generates a summary report of the numbers and types of records and jobs that were selected for inclusion in the history data set.

To run the Load utility, modify the JCL in member BSLBJLOD of data set SAMPLIB as described then submit the JCL. The steps will complete with a condition code of zero (0).

- If you did not install the MAINVIEW Batch Optimizer load modules in your MVS LNKLST, uncomment the STEPLIB and EXITLIB DD statements and verify that the data set names are correct.

- Verify that the data set name provided on the SORTLIB DD statement is correct.

- On the SELECTIN DD statement, include the name of the select file.

- Review and modify as needed the parameters that are listed after the SMFCNTL DD statement. All parameters must start in column 1.

Note: The DATE, START, END, and SID parameters are the same as those used with the SMF Dump program (IFASMFDP).

Table A-6 Load Utility Summary Report Parameters

Parameter	Description
LOAD	Must be present to load the history data set with data that is extracted from the select file. If this statement is omitted, the history data set is not populated.
SUBSYSNAME(*ssid*)	Names the MAINVIEW Batch Optimizer subsystem whose history data set you want to populate (*ssid* is a MAINVIEW Batch Optimizer subsystem ID). The subsystem procedure identifies the name of the history data set. The default subsystem ID is BCSS (as defined during the customization process).
DATE(*yyyyddd,yyyyddd*)	Select records that were recorded within the specified range of dates (*yyyy* is the year and *ddd* is the Julian date). The value for *ddd* cannot exceed 366. If DATE is specified, you must include both a start date and an end date. If DATE is not specified, the default is DATE(1900000,2099366).
START(*hhmm*)	Select records that were recorded after the START time but before the END time (*hh* is hours and *mm* is minutes based on a 24-hour clock). If START is not specified, the default is START(0000).
END(*hhmm*)	Select records that are identified with times less than, or equal to, *hhmm* (*hh* is hours and mm is minutes based on a 24-hour clock). If END is not specified, the default is END(2400).
SID(*smfname*)	Select records that were recorded by a specific operating system (*smfname* is the system identifier). SID can be any four alphanumeric characters. If you do not specify SID, records about any operating system are selected.
JOBNAME(*jobname,jobname,jobname*,...)	Select records that pertain to jobs with specific names or generic name patterns (*jobname* is a specific job name or the generic job name pattern that contains one or more asterisks (*)). You can specify up to 1,000 specifications. To continue to your specifications to another line, end the current line with a comma. You can also specify multiple JOBNAME parameters. If JOBNAME is not specified, the default is JOBNAME(*).
MINEXECTIME(*mmss*)	Do not select jobs whose elapsed time is less than he specified amount (*mm* is minutes and *ss* is seconds). If MINEXECTIME is not specified, the default is MINEXECTIME(0).
MAXHISTPCT(*xx*)	Specify how full the history data set should become with SMF records (*xx* is a number from 01 through 75). If MAXHISTPCT is not specified, the default is MAXHISTPCT(50).

As Figure A-4 shows, the summary report lists the parameters and values that were used to select records for inclusion in the history data set. The summary report also summarizes the SMF records that the utility used as input and those it selected as output.

If you entered invalid parameters, the summary report contains messages that describe the error.

Figure A-4 Summary Report from the Load Utility

```
OUTPUT SUMMARY
32 NEW JOB INSTANCES ADDED TO HISTORY DATA SET FOR SUBSYSTEM BCSS
 0 JOB INSTANCES MERGED WITH HISTORY
 0 JOB INSTANCES REPLACED IN HISTORY
 0 JOB INSTANCES NOT REPLACED IN HISTORY BECAUSE ACTUAL (NON-SMF) DATA
EXISTS
 0 HISTORY POPULATE REQUESTS NOT COMPLETED BY ACCESS METHOD
 0 JOB INSTANCES WRITTEN TO SELECT FILE
9,223 JOBS DISCARDED BECAUSE THEY DID NOT MEET THE SELECTION CRITERIA
 0 JOBS DISCARDED WITH ELAPSED TIME LESS THAN MINIMUM
 0 JOBS DISCARDED BECAUSE OF INCOMPLETE EXECUTION
```

Output Summary

The Load Utility report provides a summary of the following output records:

- NEW JOB INSTANCES ADDED TO HISTORY DATA SET FOR SUBSYSTEM *ssid*

  Number of jobs that were written to the history data set that is associated with the identified subsystem (*ssid*). These jobs did not previously appear in the history data set.

- JOB INSTANCES MERGED WITH HISTORY

  Number of jobs that match an existing job name in the history data set. The new SMF data is merged with the existing SMF or Job Optimizer data because the job structure is the same.

- JOB INSTANCES REPLACED IN HISTORY

  Number of jobs that match an existing job in the history data set. The new SMF data replaces the existing SMF data because the job structure has changed. Actual Job Optimizer data is not replaced.

- JOB INSTANCES NOT REPLACED IN HISTORY BECAUSE ACTUAL (NON-SMF) DATA EXISTS

  Number of jobs that match an existing job name in the history data set. The new SMF data is discarded because SMF data will not replace actual Job Optimizer data.

- HISTORY POPULATE NOT COMPLETED BY ACCESS METHOD

  Number of jobs for which errors occurred and, as a result, were not added to the history data set.

- JOB INSTANCES WRITTEN TO SELECT FILE

  Number of jobs that were written to the select file.

- JOBS DISCARDED BECAUSE THEY DID NOT MEET THE SELECTION CRITERIA

  Number of jobs that were not loaded to the history data set because they did not meet the selection criteria.

- JOBS DISCARDED WITH ELAPSED TIME LESS THAN MINIMUM

  Number of jobs that were not loaded to the history data set because their elapsed time was less than the specified minimum execution time.

- JOBS DISCARDED BECAUSE OF INCOMPLETE EXECUTION

  Number of jobs that were not loaded to the history data set because they did not complete successfully.

Appendix B Navigating the User Interface

This appendix describes the MAINVIEW Batch Optimizer user interface. This appendix discusses the following topics:

Overview

The interface is an ISPF-based interactive dialog that provides access to the MAINVIEW Batch Optimizer control data set. This data set is allocated to the BSLPLEX data definition (DD) statement of the MAINVIEW Batch Optimizer subsystem started task. Although you can use the ISPF Edit function to view or modify the control data set members, the MAINVIEW Batch Optimizer user interface affords a protected environment for this process. The user interface verifies that any control data set members you create or modify are syntactically correct and do not provide conflicting instructions to the optimization components.

ISPF Conventions

This section provides general information about the user interface and the panels that are provided with the dialog. All panels, pull-down menus, and pop–up windows in this interface conform to IBM CUA standards.

ISPF Profile

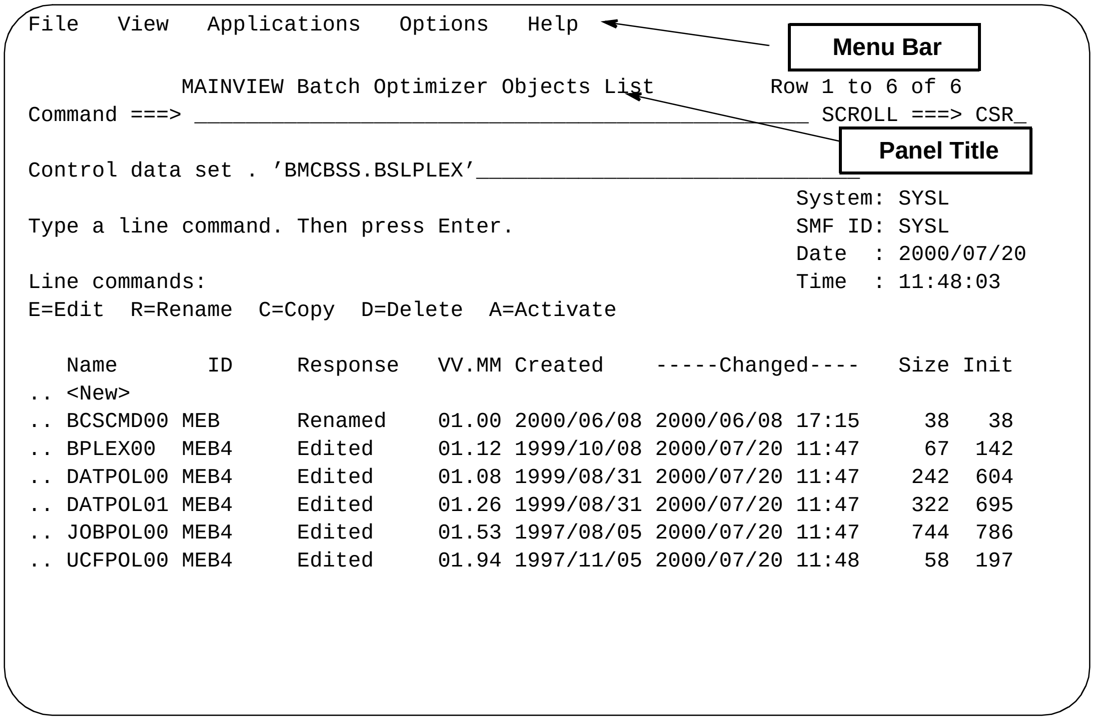
When you access the ISPF interface for the first time, the MAINVIEW Batch Optimizer dialog creates member BSSBPROF in your ISPF profile data set. This member contains information about your dialog sessions and some of the input field values from your previous session.

For example, the dialog maintains the name of the control data set that was most recently accessed. If you modified MAINVIEW Batch Optimizer dialog keylists, BSSBPROF also contains a copy of member BSSBKEYS from ISPTLIB with your modifications. Each time you access the MAINVIEW Batch Optimizer user interface, it restores values in the data entry and choice entry fields from information saved in the BSSBPROF member. If you delete member BSSBPROF from your profile data set, the MAINVIEW Batch Optimizer user interface always creates a new one for you. The new member always contains MAINVIEW Batch Optimizer default values.

Panel Layout

Figure B-1 shows a MAINVIEW Batch Optimizer Objects List panel.

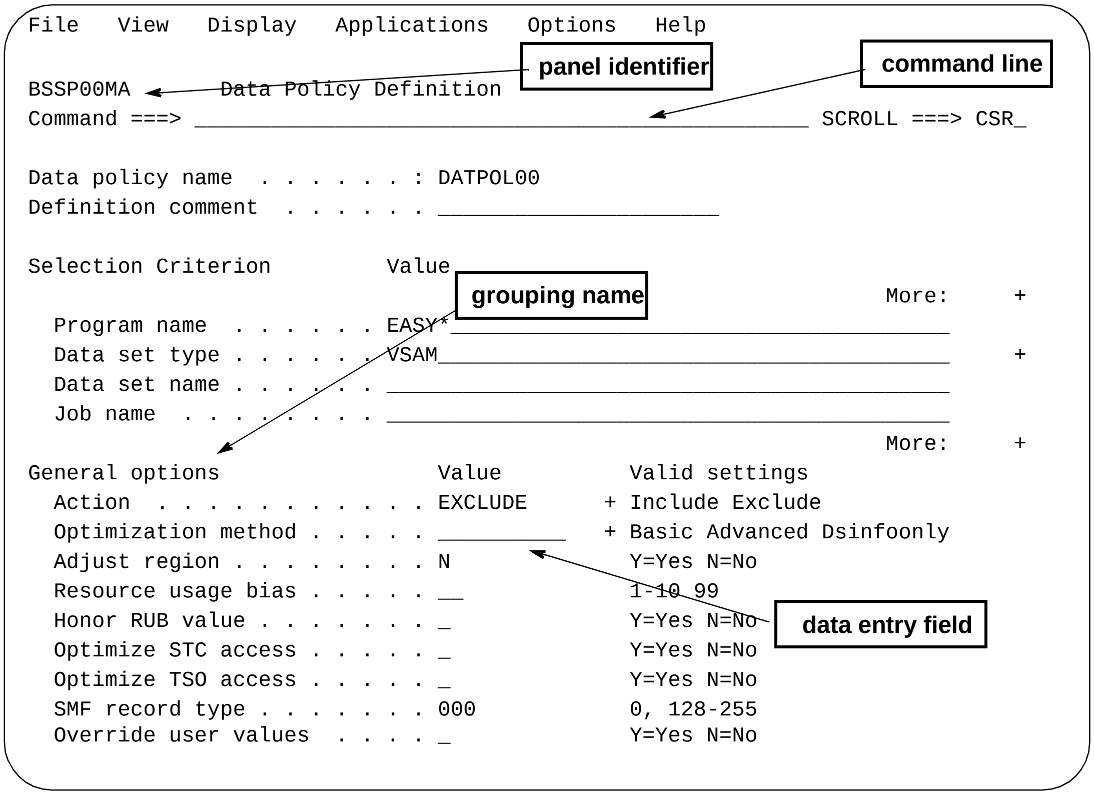
Figure B-1 MAINVIEW Batch Optimizer Objects List Panel



```
File   View   Applications   Options   Help

            MAINVIEW Batch Optimizer Objects List         Row 1 to 6 of 6
Command ===> ________________________________________________ SCROLL ===> CSR_

Control data set . 'BMCBSS.BSLPLEX'______________________________
                                                            System: SYSL
Type a line command. Then press Enter.                      SMF ID: SYSL
                                                            Date  : 2000/07/20
Line commands:                                              Time  : 11:48:03
E=Edit  R=Rename  C=Copy  D=Delete  A=Activate

   Name       ID     Response   VV.MM Created    -----Changed----   Size Init
.. <New>
.. BCSCMD00 MEB      Renamed    01.00 2000/06/08 2000/06/08 17:15     38   38
.. BPLEX00  MEB4     Edited     01.12 1999/10/08 2000/07/20 11:47     67  142
.. DATPOL00 MEB4     Edited     01.08 1999/08/31 2000/07/20 11:47    242  604
.. DATPOL01 MEB4     Edited     01.26 1999/08/31 2000/07/20 11:47    322  695
.. JOBPOL00 MEB4     Edited     01.53 1997/08/05 2000/07/20 11:47    744  786
.. UCFPOL00 MEB4     Edited     01.94 1997/11/05 2000/07/20 11:48     58  197
```

Figure B-2 shows a MAINVIEW Batch Optimizer data entry panel.

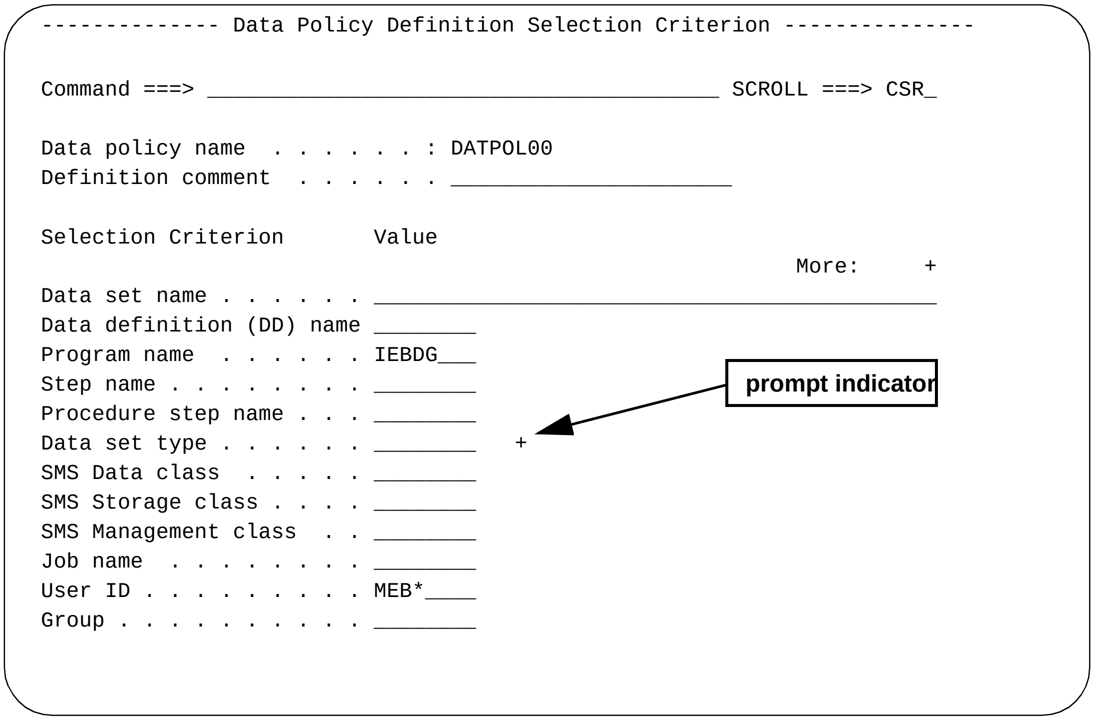
Figure B-2 Data Policy Definition Panel



```
File   View   Display   Applications   Options   Help

BSSP00MA       Data Policy Definition
Command ===> ________________________________________________ SCROLL ===> CSR_

Data policy name  . . . . . . : DATPOL00
Definition comment  . . . . . . ______________________

Selection Criterion         Value
                                                                More:     +
  Program name  . . . . . . EASY*_______________________________________
  Data set type . . . . . . VSAM________________________________________     +
  Data set name . . . . . . ____________________________________________
  Job name  . . . . . . . . ____________________________________________
                                                                More:     +
General options                 Value          Valid settings
  Action  . . . . . . . . . . . EXCLUDE      + Include Exclude
  Optimization method . . . . . __________   + Basic Advanced Dsinfoonly
  Adjust region . . . . . . . . N              Y=Yes N=No
  Resource usage bias . . . . . __             1-10 99
  Honor RUB value . . . . . . . _              Y=Yes N=No
  Optimize STC access . . . . . _              Y=Yes N=No
  Optimize TSO access . . . . . _              Y=Yes N=No
  SMF record type . . . . . . . 000            0, 128-255
  Override user values  . . . . _              Y=Yes N=No
```

Figure B-3 shows a MAINVIEW Batch Optimizer pop–up panel.

Figure B-3 Data Policy Definition Selection Criterion Pop–up Panel



```
-------------- Data Policy Definition Selection Criterion ---------------

Command ===> ________________________________________ SCROLL ===> CSR_

Data policy name  . . . . . . : DATPOL00
Definition comment  . . . . . . ______________________

Selection Criterion       Value
                                                            More:     +
Data set name . . . . . . ____________________________________________
Data definition (DD) name ________
Program name  . . . . . . IEBDG___
Step name . . . . . . . . ________
Procedure step name . . . ________
Data set type . . . . . . ________   +
SMS Data class  . . . . . ________
SMS Storage class . . . . ________
SMS Management class  . . ________
Job name  . . . . . . . . ________
User ID . . . . . . . . . MEB*____
Group . . . . . . . . . . ________
```

prompt indicator

Action Bar

Each panel (except for pop–up panels) has an action bar at the top of the panel. The selections on the action bar enable you to access pull-down menus for navigation or processing purposes. Action bar selections differ according to panel.

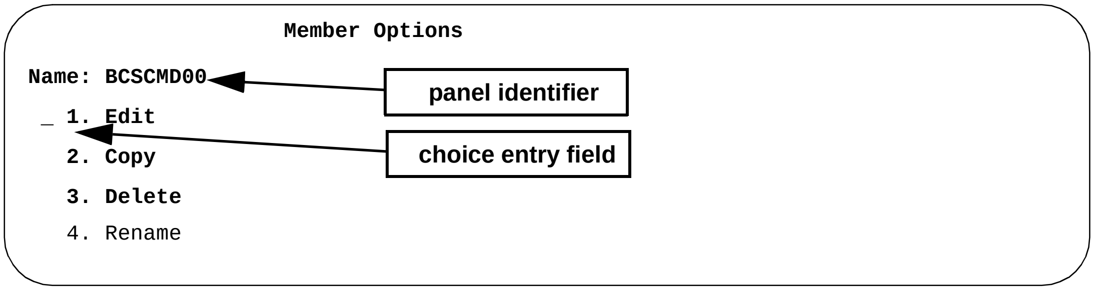
Panel Title

Each panel has a unique name that is displayed at the top of the panel.

Panel Identifier

Each panel has a unique panel identifier. The ISPF interface does not display the panel identifier when you first access the interface. The same panel samples in this manual display the panel identifiers for reference purposes. To display the identifier in the upper left corner of the panel, type **PANELID** on the command line and press **Enter**. To remove the panel identifier, type **PANELID** again and press **Enter**. Figure B-4 shows a panel that displays the panel name.

Figure B-4 Member Options Choice Entry Panel



```
                      Member Options

Name: BCSCMD00
 _ 1. Edit
   2. Copy
   3. Delete
   4. Rename
```

Command Line

You can issue TSO and ISPF commands from any panel with a command line. To issue a command, type the prefix (**TSO** or **ISPF**) and the command on the command line, and press **Enter**. For example, to obtain information about a data set, type **TSO LISTD ‘ABC.DATA.SET’** on the command line, and press **Enter**. You also can use the command line to issue a command that is equivalent to an F key. The command equivalent to F1 is HELP.

To access help, press **F1** or type **HELP** on the command line and press **Enter**. See “F keys” on page B-7 for a list of all F keys and their command equivalents. You can position the command line at the top or bottom of the panel. For instructions on repositioning the command line, see “Dialog Options and Settings” on page B-10.

Choice Entry Field

Numbered choice entry panels have a single or double character field to the left of the first option in the list. To select an option, type your selection in the choice entry field and press **Enter**. If the point-and-shoot feature is enabled, you can position the cursor to the line corresponding to your choice and press **Enter**.

Grouping Name

An option may have several fields that relate to the same option. The panel a grouping name is displayed for organizational and usability purposes.

Data Entry Field

When you can enter data for an option, a data entry field is displayed to the right of the option. This field may be blank or it may contain a default value. To enter or change a value, type a value and clear remaining characters.

Scrolling Indicator

This area of a panel indicates that more information exists outside the visible panel area and shows you the direction to scroll to see that information. If More is followed by a minus symbol, press **F7** (UP). If you see More followed by a plus symbol, press **F8** (DOWN).

Prompt Indicator

The dialog uses a plus (+) symbol to designate input fields that support the Prompt command. When the dialog appends this indicator to an input field, you can use the prompt facility to obtain a list of valid options for the field. Obtain the list by positioning your cursor in the field and pressing **F4** (Prompt) or by positioning your cursor beneath the plus symbol and pressing **Enter** (if you have enabled the point-and-shoot facility).

F keys

The MAINVIEW Batch Optimizer ISPF dialog uses F keys for navigation and processing purposes. To display the F keys, type **FKA** on the command line, and press **Enter**. The F keys appear at the bottom of each panel. MAINVIEW Batch Optimizer uses the following F key settings. The dialog stores these settings in the keylist table. See “Modifying Key Lists” on page B-12 for information about changing the F key settings. You can press the associated F key or type the equivalent command on the command line.

Note: The labels shown with the F keys on panels are not always the equivalent commands.

F1 (HELP) Display field-level, panel-level, message-level help, or reference phrase help.

F2 (SPLIT) Split the screen at the cursor position.

F3 (EXIT) Terminate the process and provide a Confirm Exit pop–up window to save or delete any changes.

F4 (PROMPT) For input fields followed by a plus (+) symbol, display a list of valid options for the input field.

F7 (UP) On a panel with a scroll bar, scroll backward in the area—toward the beginning of the entry. Use this key if you see More: -.

F8 (DOWN) On a panel with a scroll bar, scroll forward in the area—toward the end of the entry. Use this key if you see More: +.

F9 (SWAP) Toggle between ISPF screens.

F12 (CANCEL) Discard any changes made on the panel, and return to the preceding panel.

Messages As you navigate through the MAINVIEW Batch Optimizer panels, the dialogs may display messages that provide information about an action that has occurred. The message could be an informational message to explain that a process has completed successfully or an error message to explain that you have entered invalid data. The dialog displays these messages in the form of short messages or pop–up messages. When you receive a short message, you can press **F1** (HELP) to view a more descriptive message (a long message). Pop–up messages usually appear at the bottom of the panel, covering your F key selections. To see the F key lines, use the ISPF Window service to move the pop–up or press **F12** (CANCEL) to remove the message.

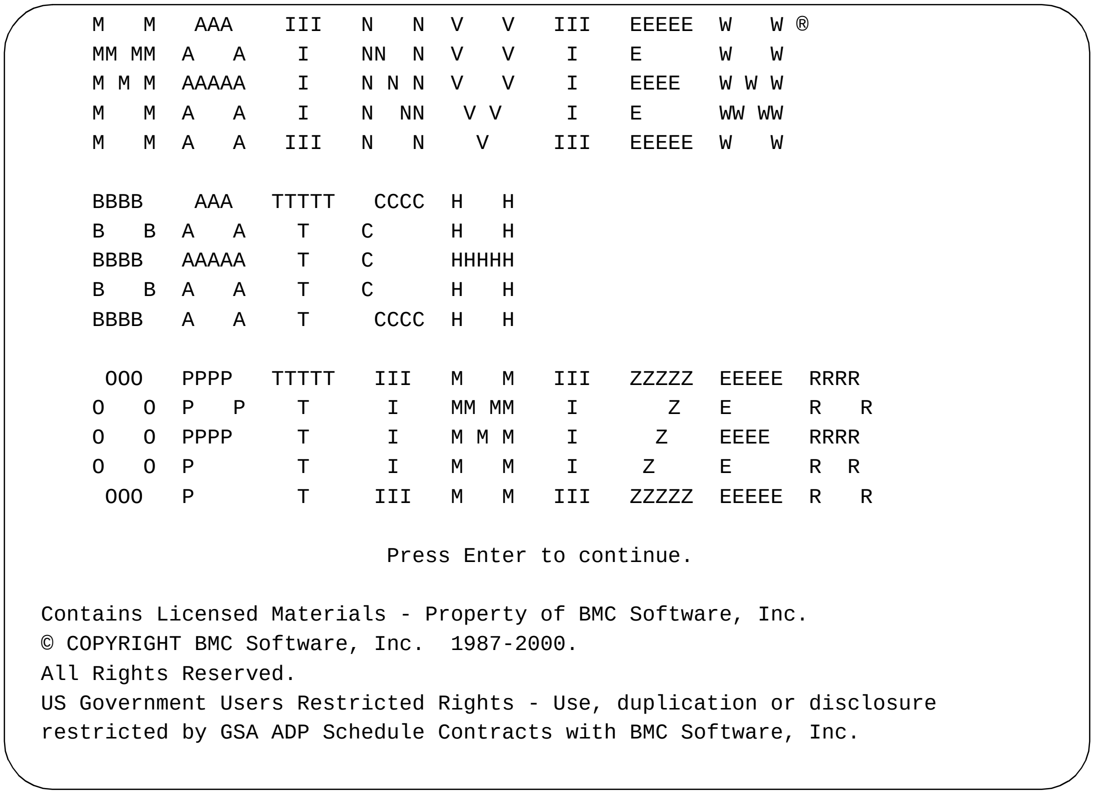
Accessing the User Interface

You can access the MAINVIEW Batch Optimizer user interface only if you have installed MAINVIEW Batch Optimizer successfully and have performed the operation verification procedure (see the *MAINVIEW Batch Optimizer Release Notes*).

To access the MAINVIEW Batch Optimizer user interface, complete the following steps:

Step 1 *Required.* Use the REXX Exec or menu selection established during installation to access the MAINVIEW Batch Optimizer customer interface. You can find a sample REXX Exec designed to access the dialog in member BSSCISPF in data set BMC.BSS.INSTALL. Figure B-5 shows the MAINVIEW Batch Optimizer Logon panel.

Figure B-5 MAINVIEW Batch Optimizer Logon panel



```
    M   M   AAA    III   N   N  V   V   III   EEEEE  W   W ®
    MM MM  A   A    I    NN  N  V   V    I    E      W   W
    M M M  AAAAA    I    N N N  V   V    I    EEEE   W W W
    M   M  A   A    I    N   NN  V V     I    E      WW WW
    M   M  A   A   III   N   N    V     III   EEEEE  W   W

    BBBB    AAA   TTTTT   CCCC  H   H
    B   B  A   A    T    C      H   H
    BBBB   AAAAA    T    C      HHHHH
    B   B  A   A    T    C      H   H
    BBBB   A   A    T     CCCC  H   H

     OOO   PPPP   TTTTT   III   M   M   III   ZZZZZ  EEEEE  RRRR
    O   O  P   P    T      I    MM MM    I      Z   E      R   R
    O   O  PPPP     T      I    M M M    I     Z    EEEE   RRRR
    O   O  P        T      I    M   M    I    Z     E      R  R
     OOO   P        T     III   M   M   III   ZZZZZ  EEEEE  R   R

                           Press Enter to continue.

Contains Licensed Materials - Property of BMC Software, Inc.
© COPYRIGHT BMC Software, Inc.  1987-2000.
All Rights Reserved.
US Government Users Restricted Rights - Use, duplication or disclosure
restricted by GSA ADP Schedule Contracts with BMC Software, Inc.
```

The logo panel contains trademark and copyright information about BMC Software and MAINVIEW Batch Optimizer. The logo panel appears each time you access the dialog unless you disable this facility. See “Dialog Options and Settings” on page B-10 for instructions about disabling the logo panel.

Step 2 *Required*. Press **Enter** to continue. The dialog displays the MAINVIEW Batch Optimizer Objects List. You also can select **F3** (EXIT/END) or **F12** (CANCEL) to leave the ISPF interface.

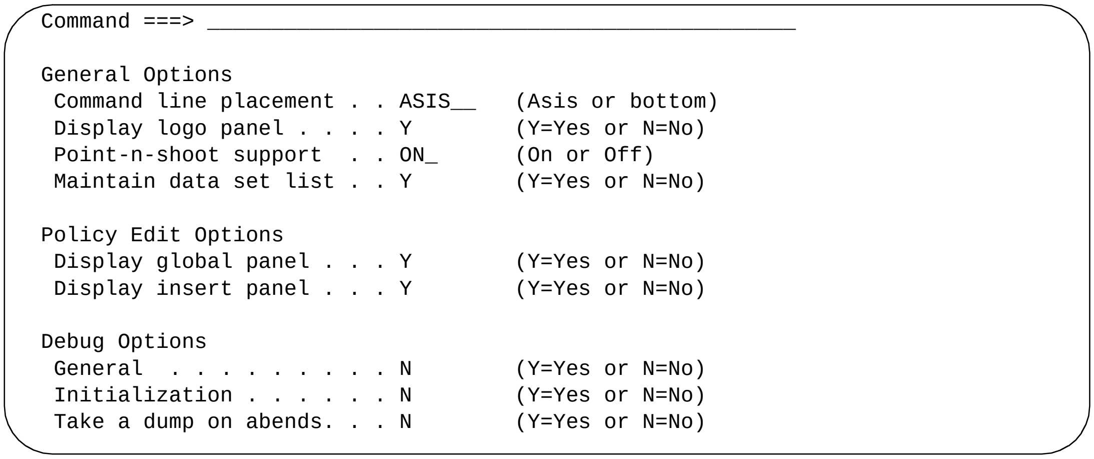
Modifying User Interface Processing

You can modify certain aspects of the dialog’s behavior by using dialog specific options, ISPF options, or Program Function Key (PFK) changes via Keylist modifications. Some of the modifications available in the dialog include enabling/disabling the point-and-shoot facility and cursor positioning.

Dialog Options and Settings

You can access the dialog options by selecting the Dialog pull-down choice of the Options action bar item or by entering the command **DLGOPTNS** on the command line. Figure B-6 shows the Dialog Options panel.

Figure B-6 Dialog Options Panel



```
Command ===> ______________________________________________

General Options
 Command line placement . . ASIS__   (Asis or bottom)
 Display logo panel . . . . Y        (Y=Yes or N=No)
 Point-n-shoot support  . . ON_      (On or Off)
 Maintain data set list . . Y        (Y=Yes or N=No)

Policy Edit Options
 Display global panel . . . Y        (Y=Yes or N=No)
 Display insert panel . . . Y        (Y=Yes or N=No)

Debug Options
 General  . . . . . . . . . N        (Y=Yes or N=No)
 Initialization . . . . . . N        (Y=Yes or N=No)
 Take a dump on abends. . . N        (Y=Yes or N=No)
```

General Options

General Options lets you to control dialog functions such as command line location or displaying the logo panel on entry. You can also choose to disable the point-and-shoot facility. You may want to disable the point-and-shoot facility if you are not using a terminal emulator with mouse support. ISPF treats point-and-shoot fields as “input” fields, enabling this options increases the amount of tabbing required to position the cursor.

If you specify a “Y” on the “Maintain data set list” option, the dialog will maintain a list of each control data set you reference. You can access this list by using the Prompt facility for the Control data set input field on the MAINVIEW Batch Optimizer Objects List panel. The dialog maintains this list in an ISPF table named BSSBDSNT that is stored in the data set allocated to your ISPTABL file. If the ISPTABL file is unavailable to your TSO session, the dialog issues a warning message and disables this facility.

Policy Edit Options

The Policy Edit Options let you to control the amount of assistance the dialog will offer when creating new control data set members. When creating new members, there are logical steps to perform. By enabling the Display global panel, when you elect to create a new control data set member, the dialog will first display the “global specifications” panel for the member. You can modify or accept the displayed values and press **Enter** or enter **F12** (Cancel) to proceed. By enabling the Display insert panel, when you elect to create a new control data set member, the dialog will first display the “statement insert” panel for the member. If you enable both options, the dialog first displays the “global specifications” panel followed by the “statement insert” panel. If you elect to disable both of these options, when you create a new control data set member, the dialog will place you in an empty table list for the member. This option is useful for those unfamiliar with the requirements of the various control data set members.

Note: These options have no effect on processing the BCSS Commands member.

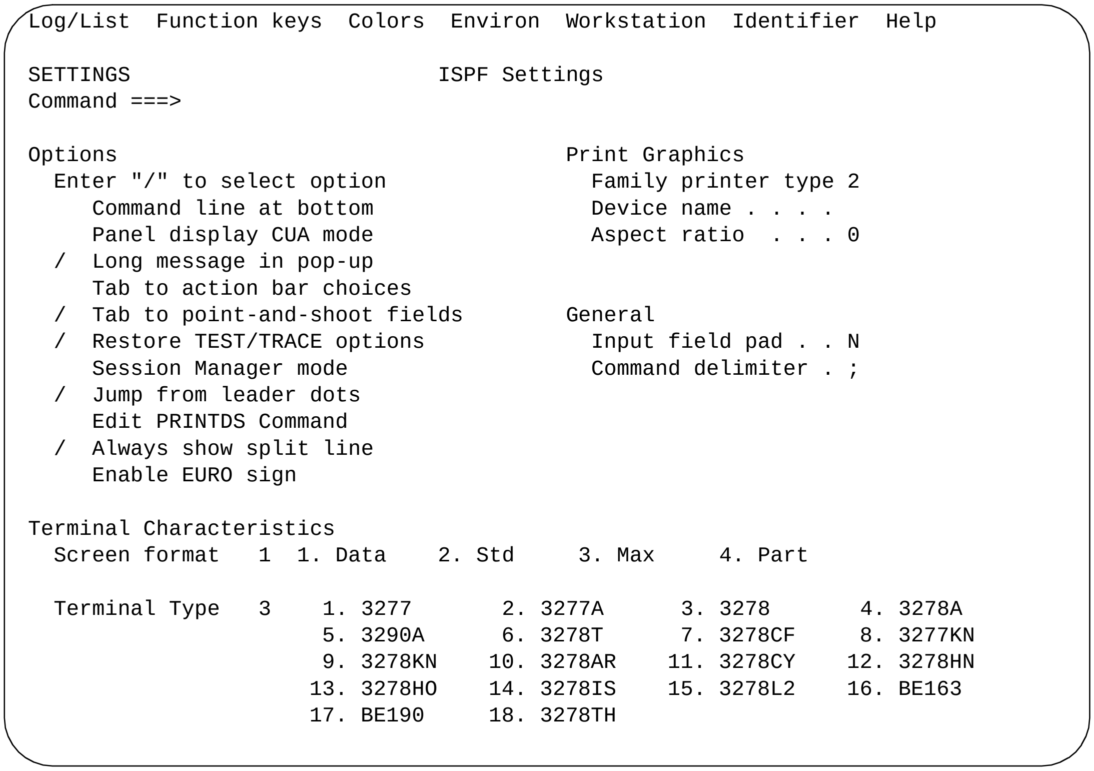
Debug Options

Leave these options set to “N” unless instructed otherwise by BMC Software Product Support.

Modifying ISPF Settings

You can access the ISPF setting by selecting the ISPF pull-down choice of the Options action bar item or by entering the command **SETTINGS** on the command line. Figure B-7 shows the ISPF Settings panel.

Figure B-7 ISPF Settings Panel



```
Log/List  Function keys  Colors  Environ  Workstation  Identifier  Help

SETTINGS                        ISPF Settings
Command ===>

Options                                  Print Graphics
  Enter "/" to select option               Family printer type 2
     Command line at bottom                Device name . . . .
     Panel display CUA mode                Aspect ratio  . . . 0
  /  Long message in pop-up
     Tab to action bar choices
  /  Tab to point-and-shoot fields       General
  /  Restore TEST/TRACE options            Input field pad . . N
     Session Manager mode                  Command delimiter . ;
  /  Jump from leader dots
     Edit PRINTDS Command
  /  Always show split line
     Enable EURO sign

Terminal Characteristics
  Screen format   1  1. Data    2. Std     3. Max     4. Part

  Terminal Type   3    1. 3277       2. 3277A      3. 3278       4. 3278A
                       5. 3290A      6. 3278T      7. 3278CF     8. 3277KN
                       9. 3278KN    10. 3278AR    11. 3278CY    12. 3278HN
                      13. 3278HO    14. 3278IS    15. 3278L2    16. BE163
                      17. BE190     18. 3278TH
```

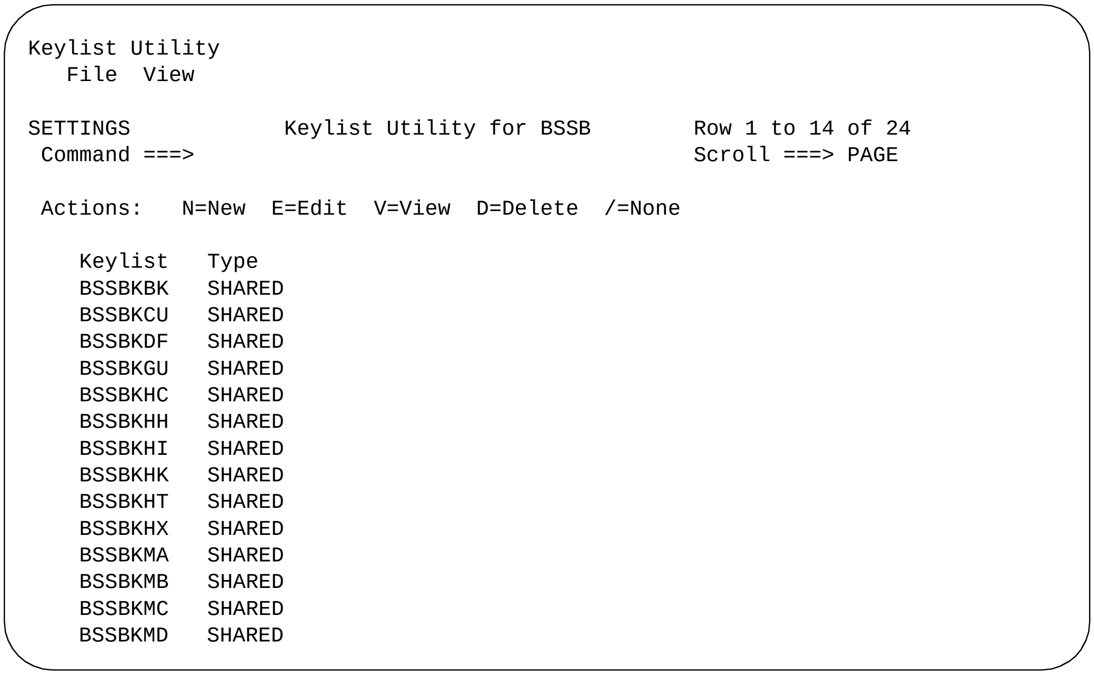
This ISPF panel lets you alter your ISPF session defaults. The MAINVIEW Batch Optimizer user interface offers direct access to ISPF settings so that you can enable or disable the “Tab to action bar choices” or “Tab to point-and-shoot fields” features. You can use the “Command line at bottom” option or the “Long message in pop-up” option to change the command line or message placement, respectively.

Note: Changes made on the ISPF Settings dialog apply to your ISPF session and not only your MAINVIEW Batch Optimizer dialog session.

Modifying Key Lists

You can access the key lists by selecting the Keylists pull-down choice of the Options action bar item or by entering the command **KEYLIST** on the command line. Figure B-8 shows the Keylist Utility panel.

Figure B-8 Keylist Utility Panel



```
Keylist Utility
   File  View

SETTINGS            Keylist Utility for BSSB        Row 1 to 14 of 24
 Command ===>                                       Scroll ===> PAGE

 Actions:   N=New  E=Edit  V=View  D=Delete  /=None

    Keylist   Type
    BSSBKBK   SHARED
    BSSBKCU   SHARED
    BSSBKDF   SHARED
    BSSBKGU   SHARED
    BSSBKHC   SHARED
    BSSBKHH   SHARED
    BSSBKHI   SHARED
    BSSBKHK   SHARED
    BSSBKHT   SHARED
    BSSBKHX   SHARED
    BSSBKMA   SHARED
    BSSBKMB   SHARED
    BSSBKMC   SHARED
    BSSBKMD   SHARED
```

Warning! BMC Software recommends that you *do not* modify the keylists. When you modify the keylists, MAINVIEW Batch Optimizer copies the content of member BSSBKEYS from your ISPTLIB library and places this information, with your changes, into member BSSBPROF of your ISPF profile data set. From the time you modify the keylists until you delete member BSSBPROF, you will access your personal copy of the keylists. If MAINVIEW Batch Optimizer changes any keylists in a maintenance release, you will not have access to these changes, which could results in unpredictable results while using the MAINVIEW Batch Optimizer user interface.

Using the User Interface

This section provides general instructions on how to use the user interface. It discusses navigation of the MAINVIEW Batch Optimizer user interface using common ISPF commands and facilities, mechanisms for handling the display of pop-up messages and data entry panels, and different shortcuts provided by the dialog as well as ISPF. See the *ISPF Users Guide* for a detailed discussion of ISPF commands and facilities.

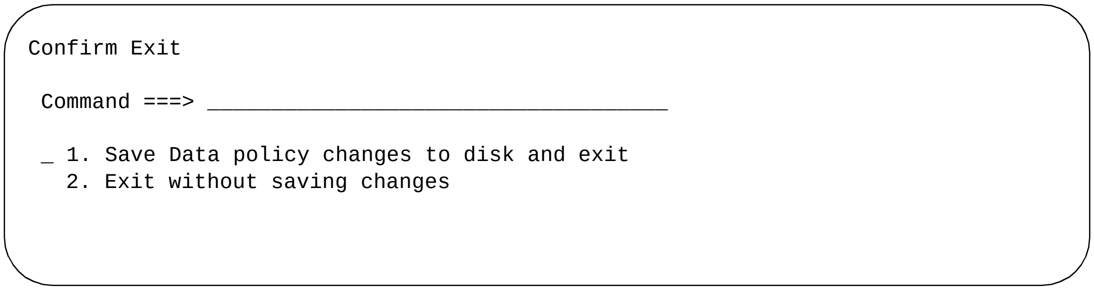
Navigation

Navigate user interface panels with **F3** (EXIT or END), **F12** (CANCEL), or **Enter** keys.

Exit or End

EXIT or END returns you to the next higher level application or exits the dialog. For example, if you select a Data Policy for editing and issue the EXIT command, the dialog returns to the MAINVIEW Batch Optimizer Objects List panel. If you changed the policy member, the dialog prompts you for conformation by displaying the pop-up panel before exiting, as shown in Figure B-9.

Figure B-9 Exit Confirmation Panel

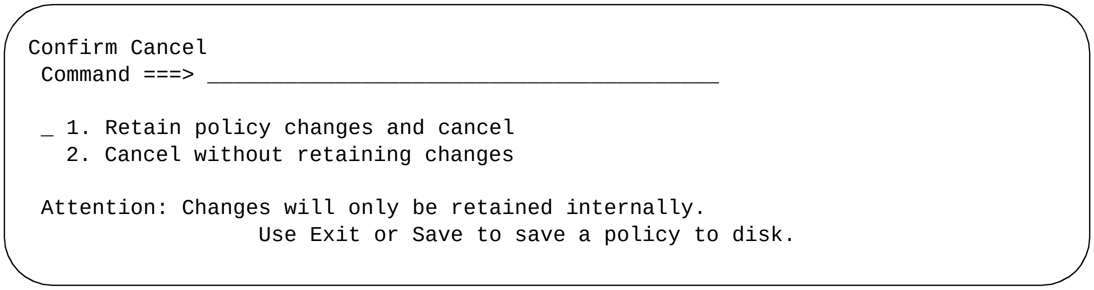


```
Confirm Exit

 Command ===> ____________________________________

 _ 1. Save Data policy changes to disk and exit
   2. Exit without saving changes
```

Cancel

CANCEL returns you to the previously displayed panel. If the prior panel was part of a higher level application, then you will exit the current application. If you enter CANCEL after making changes to a data entry panel, the dialog prompts you for confirmation. Figure B-10 on page B-15 shows a Cancel Confirmation panel.

Figure B-10 Cancel Confirmation Panel



```
Confirm Cancel
 Command ===> ________________________________________

 _ 1. Retain policy changes and cancel
   2. Cancel without retaining changes

 Attention: Changes will only be retained internally.
                  Use Exit or Save to save a policy to disk.
```

Enter

Use the **Enter** key to move forward. Either commit data entry panel changes, invoke a command, select a line item from a list (or table) of items, or select one of several numbered choices on a choice entry panel with the **Enter** key.

Pop-up Panels

The MAINVIEW Batch Optimizer user interface makes extensive use of pop-up panels for data entry and messages. The dialog places pop-up error messages near the field in error. It places pop-up prompt lists near the input field. It is best to view the dialog on a screen capable of displaying 43 lines of data (a 3270 Model 4 terminal or terminal emulator). When using a smaller screen size, pop-up panels may obscure underlying data necessary to correct erroneous input. ISPF offers the following methods for addressing this issue:

- ISPF Window command
- ISPF Resize command

ISPF Window Command

The ISPF Window command allows you to move an active pop-up to another location within the screen space. Use this command to expose underlying fields obscured by the pop-up. There are two methods of invoking this command. One method is to type **WINDOW** at the command line (or use a function key), move the cursor to the position where you want the pop-up to appear, and press **Enter**. Alternatively, place the cursor anywhere on the active pop-up frame and press **Enter,** (ISPF will issue the message WINDOW MOVE PENDING). Move the cursor to the position where you want the pop–up to appear, and press **Enter** again.

See the IBM *ISPF Users Guide* or the ISPF interactive help facility for more details on this command.

ISPF Resize Command

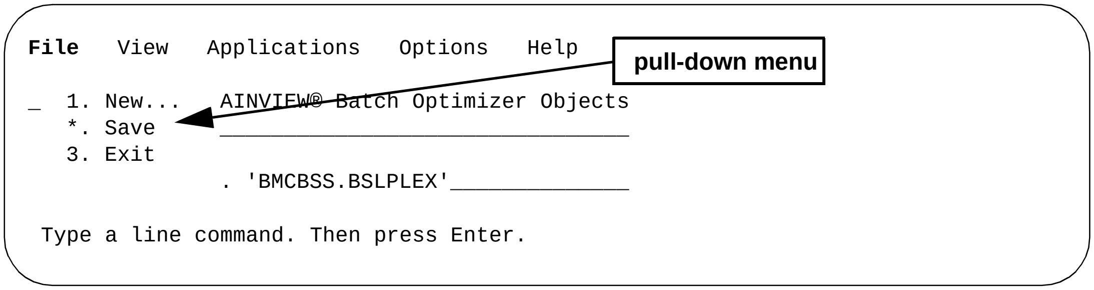
The MAINVIEW Batch Optimizer user interface limits the line depth of a pop-up screen to 20 lines. Although the actual size of a panel may exceed this, you can only view 20 lines at a time. For pop-up screens that exceed the maximum number of lines, ISPF activates the scrolling facility (indicated by More: +) which allows you to scroll through the entire panel content. To view the entire pop-up, use the RESIZE command. The RESIZE command causes the pop–up to be displayed in the entire screen space. To restore the pop-up to its original size, reenter the RESIZE command at the command line.

See the IBM *ISPF Users Guide* or the ISPF interactive help facility for more details on this command.

Accessing the Action Bar

The Action bar offers access to the MAINVIEW Batch Optimizer user interface commands. The action bar is most accessible if you use a 3270 emulator that has mouse support for host sessions. If you do not have this support, the ISPF Actions command affords direct access to the Action bar options. To access an Action bar option by using the ISPF Actions command, type **ACTIONS** at the command line followed by the “underlined” character of the Action bar option. For example, to access the File action bar option, type **ACTIONS F**. Figure B-11 on page B-17 shows a File Pull-Down Menu panel. On panels with an Action bar, the dialog sets **F10** to Actions. This lets you to enter the “underlined” character at the command line and enter **F10** to invoke the Actions command.

Figure B-11 File Pull-Down Menu



```
File   View   Applications   Options   Help

_  1. New...   AINVIEW® Batch Optimizer Objects
   *. Save     _________________________________
   3. Exit
               . 'BMCBSS.BSLPLEX'______________

 Type a line command. Then press Enter.
```

See the IBM *ISPF Users Guide* or the ISPF interactive help facility for more details on this command.

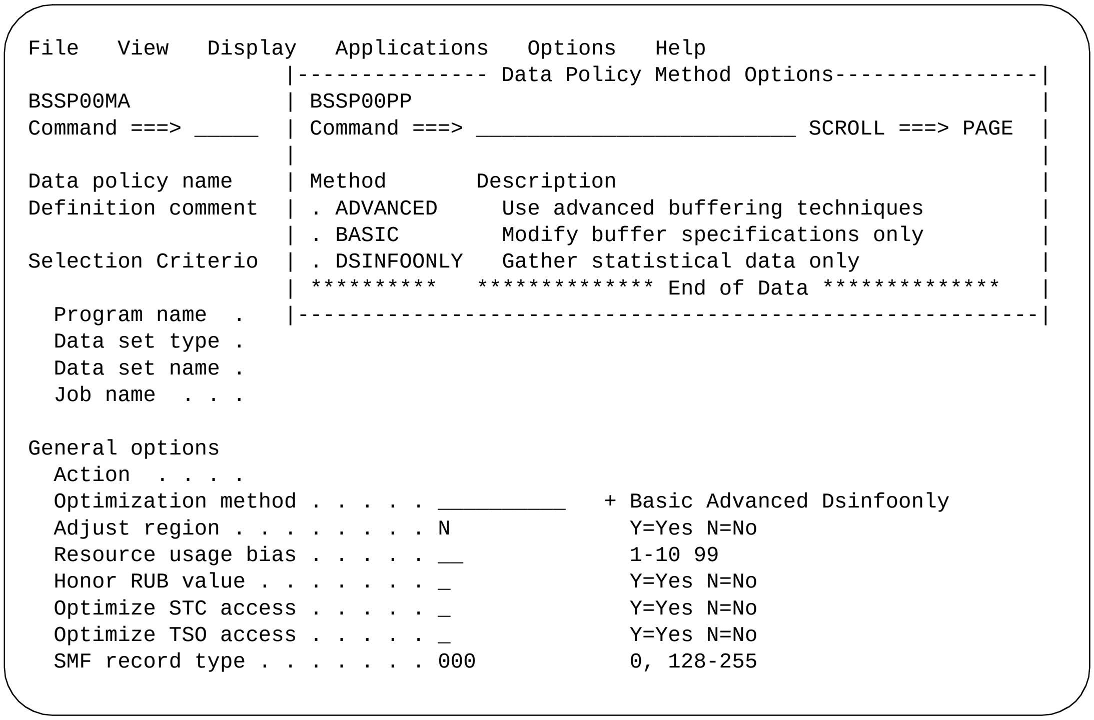
Point-and-Shoot

The MAINVIEW Batch Optimizer user interface makes extensive use of the ISPF Point-and-Shoot facility. When enabled, you can access this facility by positioning your cursor beneath a Point-and-Shoot defined field and pressing **Enter**. If your terminal emulator offers mouse support for host sessions, access this facility by moving your mouse pointer to the Point-and-Shoot field and double-click the left mouse button.

What the dialog displays in response to a point-and-shoot request depends upon the field. If you point-and-click a control data set member name on the MAINVIEW Batch Optimizer Objects List panel, the dialog displays Figure B-4 on page B-6. If you point-and-click a prompt indicator, the dialog displays a list of valid options for the input field. For example, if while editing a Data Policy definition, you click the prompt indicator adjacent to the Optimization method input field, the dialog displays the panel shown in Figure B-12 on page B-18.

You also can access the point-and-shoot facility within the prompt list to select your choice. When enabled, the point-and-shoot facility allows you to navigate through the dialog with only the **Tab** and **Enter** keys. With a 3270 emulator enabled for host mouse support, you can navigate without using the keyboard until you have to perform a data entry task.

Figure B-12 Data Policy Method Options Pop–up List Panel



```
File   View   Display   Applications   Options   Help
                    |-------------- Data Policy Method Options---------------|
BSSP00MA            | BSSP00PP                                               |
Command ===> _____  | Command ===> _________________________ SCROLL ===> PAGE  |
                    |                                                        |
Data policy name    | Method      Description                                |
Definition comment  | . ADVANCED    Use advanced buffering techniques        |
                    | . BASIC       Modify buffer specifications only        |
Selection Criterio  | . DSINFOONLY  Gather statistical data only             |
                    | **********  ************** End of Data **************  |
  Program name  .   |--------------------------------------------------------|
  Data set type .
  Data set name .
  Job name  . . .

General options
  Action  . . . .
  Optimization method . . . . . __________   + Basic Advanced Dsinfoonly
  Adjust region . . . . . . . . N              Y=Yes N=No
  Resource usage bias . . . . . __             1-10 99
  Honor RUB value . . . . . . . _              Y=Yes N=No
  Optimize STC access . . . . . _              Y=Yes N=No
  Optimize TSO access . . . . . _              Y=Yes N=No
  SMF record type . . . . . . . 000            0, 128-255
```

Creating and Editing Control Data Set Members

You use the MAINVIEW Batch Optimizer user interface to create and edit objects. These objects may be control data set members or definition statements within a control data set member. In either case, the dialog provides multiple methods for creating and editing objects.

New Command

The dialog provides a number of ways to create new objects. If you wish to create a new control data set member object, use the NEW command. The dialog provides several methods for accessing this command. You can access this command by using the NEW pull-down choice of the File action bar option, or you can type the command directly at the command line. In cases where the dialog displays a list of objects, to create a new object you can select the <NEW> line with the **E** line command or position your cursor beneath this text and press **Enter**. When the dialog displays a list of objects, it displays this *special* line at the top of the list.

Note: The <NEW> line typically does not scroll with the data in the list so that it always appears at the top of the list. The exception to this rule is certain prompt list pop–up panels.

Insert Line Command

To create new objects within a control data set member, you can use the **NEW** command or you can use the **I** line command to insert new objects at specific positions. When you use the **NEW** command to create new objects, the dialog always inserts them at the top of the list. The insert line command gives you the flexibility of choosing where a new object will appear in the list. The insert facility is available when editing policy control data set members where the position of a statement in the list is important to how the product processes the policy. The insert facility is not available for object lists where position in the list is unimportant; typically, sorted lists such as the BatchPlex Definition MVS images list or the MAINVIEW Batch Optimizer Objects List.

To use the insert line command, place an **I** on the line after which you want to place the new object and press **Enter**. The dialog will then display the appropriate data entry panel. After pressing **Enter** to submit the request, the dialog places the new object after the selected line and positions the cursor to the newly inserted line.

Copy Processing – Named Objects

The dialog allows you to copy control data set members and BatchPlex definition MVS images when creating new members. These two lists are examples of sorted lists that are insensitive to position. To copy a member or MVS image, place a **C** in front of the object you wish to copy and press **Enter**. The dialog displays a data entry panel where you can specify the name of the new object. After completing the data entry process and pressing **Enter**, the dialog inserts a new object in the correct sort order (alphabetic for control data set members and MVS images).

Rename Processing – Named Objects

The dialog allows you to rename objects that have names; such as control data set member names or MVS images. To rename a member or MVS image, type **R** in front of the object you wish to rename and press **Enter**. The dialog will display a data entry panel where you can specify the new name of the object. After completing the data entry process and pressing **Enter**, the dialog renames the object and inserts it in the correct sort order (alphabetic for control data set members and MVS images).

Copy and Paste Processing – Positional Objects

Another mechanism for creating new objects is through the Copy and Paste line commands, **C** and **P** respectively. When displaying a list where both line commands are available, you can use the copy line command to create a copy of an object by placing a **C** in front of the object you wish to copy. The dialog copies this object to a clipboard. Use the paste line command by placing a **P** on the line after which you want to place the contents of the clipboard and press **Enter**. The dialog copies the contents of the clipboard after the selected line and positions the cursor to the newly pasted line.

Cut and Paste Processing – Positional Objects

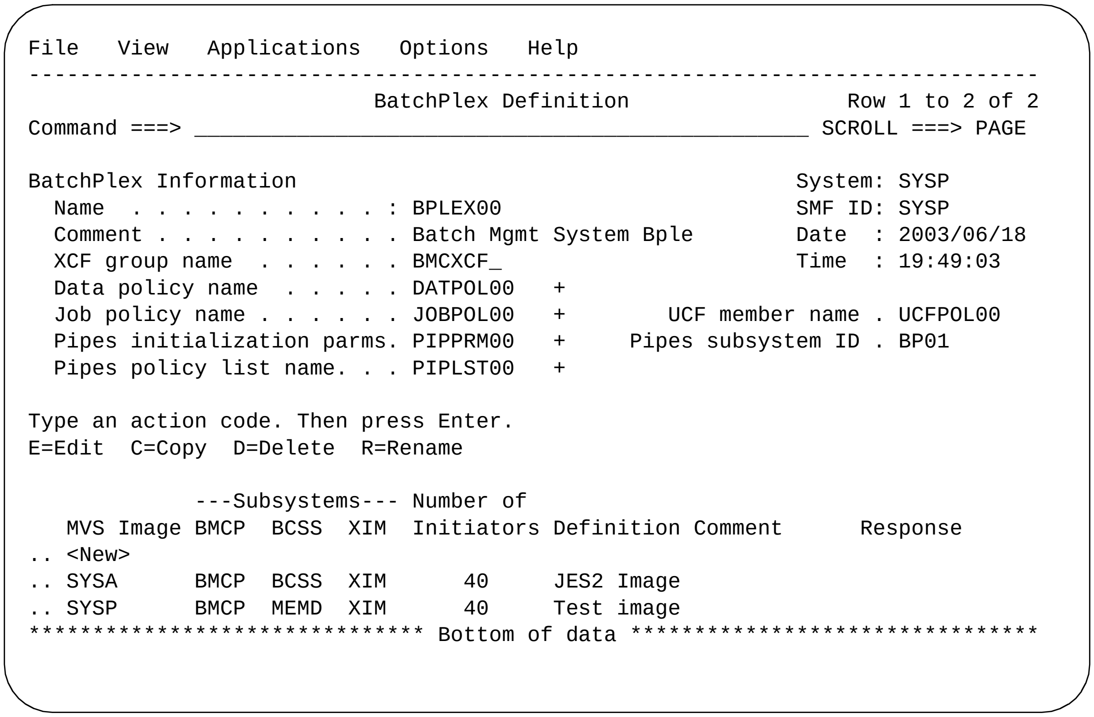
When available, the Cut and Paste line commands, **X** and **P** respectively, perform in the same manner as the COPY and PASTE processing with one exception. When you cut an object from the list using the **X** line command, the dialog removes the object from the list and places it on the clipboard. The object remains in the clipboard until you paste it at the desired location.

Warning! If you save the control data set member without pasting the object back into the list, you effectively delete the object from the list.

Object List Displays

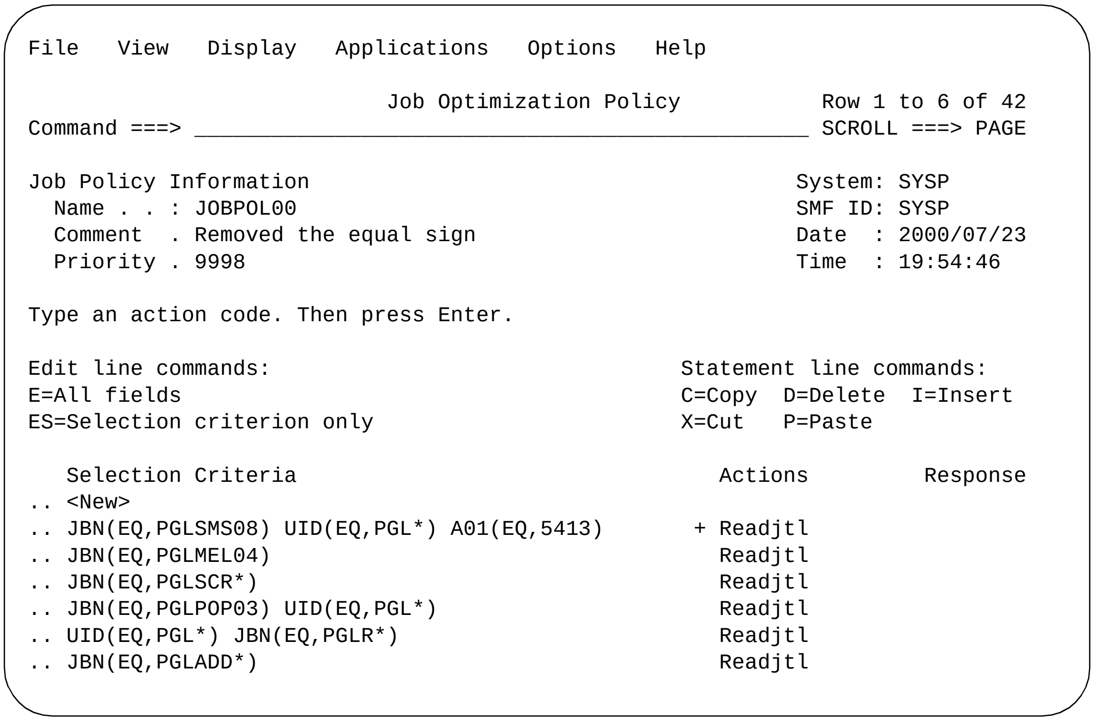
The data displayed in an object list depends on the object type. For the MAINVIEW Batch Optimizer Objects List, the data displayed is a member list (see Figure B-1 on page B-3). For the BatchPlex Definition list, the data displayed is a list of MVS images defined to the BatchPlex (see Figure B-13 on page B-21). For a Job Policy list, the data displayed is a list of selection criteria and actions to perform (see Figure B-14 on page B-21). Although each list differs in content, certain concepts remain consistent for all lists. Enter line commands to the left of the line that you wish to select. The dialog displays ”last activity” information regarding each line in a Response column. The panel displays the valid line commands for the list.

Figure B-13 BatchPlex Definition Panel



```
File   View   Applications   Options   Help
------------------------------------------------------------------------------
                           BatchPlex Definition                Row 1 to 2 of 2
Command ===> ________________________________________________ SCROLL ===> PAGE

BatchPlex Information                                         System: SYSP
  Name  . . . . . . . . . . . : BPLEX00                       SMF ID: SYSP
  Comment . . . . . . . . . . . Batch Mgmt System Bple        Date  : 2003/06/18
  XCF group name  . . . . . . . BMCXCF_                       Time  : 19:49:03
  Data policy name  . . . . . . DATPOL00   +
  Job policy name . . . . . . . JOBPOL00   +        UCF member name . UCFPOL00
  Pipes initialization parms. PIPPRM00   +     Pipes subsystem ID . BP01
  Pipes policy list name. . . PIPLST00   +

Type an action code. Then press Enter.
E=Edit  C=Copy  D=Delete  R=Rename

             ---Subsystems--- Number of
   MVS Image BMCP  BCSS  XIM  Initiators Definition Comment      Response
.. <New>
.. SYSA      BMCP  BCSS  XIM      40     JES2 Image
.. SYSP      BMCP  MEMD  XIM      40     Test image
******************************* Bottom of data ********************************
```

Figure B-14 Job Optimizer Policy Selection Criteria and Action List



```
File   View   Display   Applications   Options   Help

                            Job Optimization Policy           Row 1 to 6 of 42
Command ===> ________________________________________________ SCROLL ===> PAGE

Job Policy Information                                        System: SYSP
  Name . . : JOBPOL00                                         SMF ID: SYSP
  Comment  . Removed the equal sign                           Date  : 2000/07/23
  Priority . 9998                                             Time  : 19:54:46

Type an action code. Then press Enter.

Edit line commands:                                  Statement line commands:
E=All fields                                         C=Copy  D=Delete  I=Insert
ES=Selection criterion only                          X=Cut   P=Paste

   Selection Criteria                                   Actions         Response
.. <New>
.. JBN(EQ,PGLSMS08) UID(EQ,PGL*) A01(EQ,5413)       + Readjtl
.. JBN(EQ,PGLMEL04)                                     Readjtl
.. JBN(EQ,PGLSCR*)                                      Readjtl
.. JBN(EQ,PGLPOP03) UID(EQ,PGL*)                        Readjtl
.. UID(EQ,PGL*) JBN(EQ,PGLR*)                           Readjtl
.. JBN(EQ,PGLADD*)                                      Readjtl
```

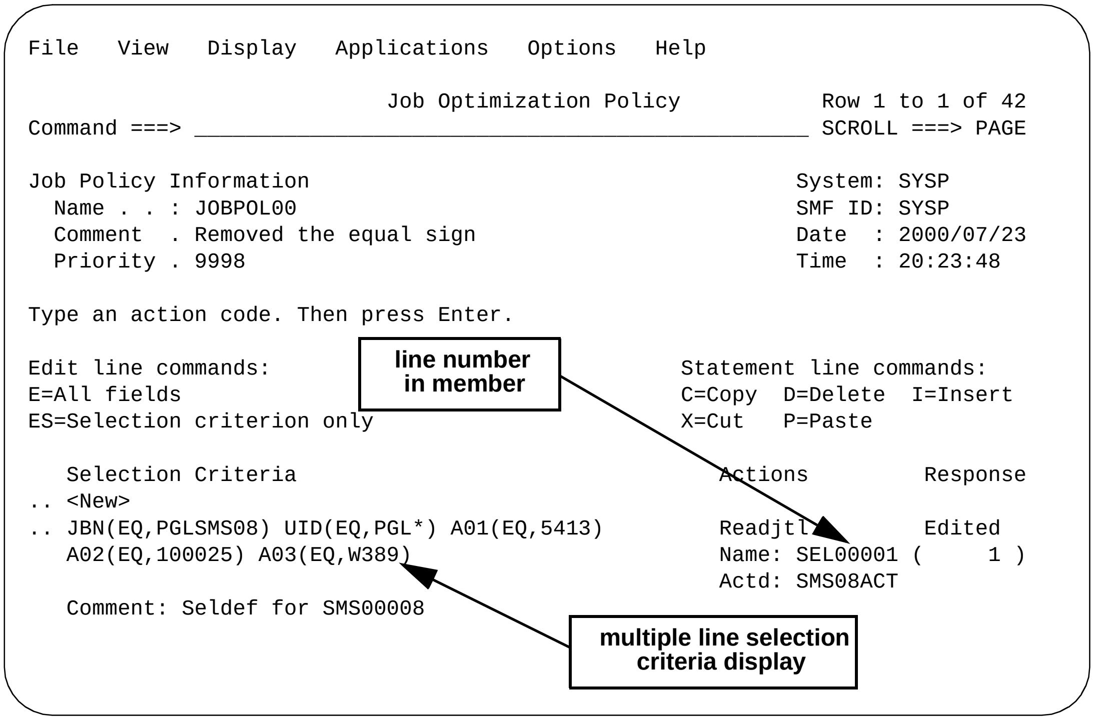
Prompt for Valid Line Command

When the dialog displays a list, the key portion of the list (typically highlighted) is defined as a point-and-shoot field. Instead of entering a line command, you can position your cursor beneath the key portion of the line and press **Enter**. The dialog will display a numbered list of valid line commands for the selected line (see Figure B-4 on page B-6). You can select a number that corresponds to the line command that you wish to invoke or position the cursor beneath the line command you want and press **Enter**. The dialog invokes the chosen line command.

Expand and Collapse

With some object lists, the dialog offers the ability to increase (Expand) or decrease (Collapse) the amount of data displayed for each item in the list. Typically, a list contains one line per item in the list. However, most list objects contain a lot more information than can be displayed in one line. For example, the Job Optimization Policy list contains a column that describes selection criteria. However, some selection criteria are too extensive to display in the allotted space. The dialog places a plus (+) symbol next to selection criteria meeting this condition (see Figure B-14 on page B-21). In such cases, you can *expand* the display by selecting the Expand Rows pull-down choice of the Display action bar option. After selecting this option, the list now displays the complete selection criteria for each object as well as diagnostic information such as the object position within the control data set member (see Figure B-15 on page B-23). Use the Collapse Rows pull-down choice of the Display action bar option to reverse the effects of the expand request.

Figure B-15 Job Optimization Policy Multi-line Criteria Display Panel



```
File   View   Display   Applications   Options   Help

                            Job Optimization Policy           Row 1 to 1 of 42
Command ===> ________________________________________________ SCROLL ===> PAGE

Job Policy Information                                      System: SYSP
  Name . . : JOBPOL00                                       SMF ID: SYSP
  Comment  . Removed the equal sign                         Date  : 2000/07/23
  Priority . 9998                                           Time  : 20:23:48

Type an action code. Then press Enter.

Edit line commands:                                Statement line commands:
E=All fields                                       C=Copy  D=Delete  I=Insert
ES=Selection criterion only                        X=Cut   P=Paste

   Selection Criteria                                 Actions         Response
.. <New>
.. JBN(EQ,PGLSMS08) UID(EQ,PGL*) A01(EQ,5413)         Readjtl         Edited
   A02(EQ,100025) A03(EQ,W389)                        Name: SEL00001 (     1 )
                                                      Actd: SMS08ACT
   Comment: Seldef for SMS00008
```

Row Update Processing

When editing some policy members, you might notice an Update action bar option. When present, this option allows you to enable *power-editing* fields in the list objects. To turn this option on, select the Row update ON pull-down choice of the Update action bar option. Once enabled, you can change fields appearing within the list display portion of the panel as input fields by over-typing them with new data. Once you press **Enter**, the dialog updates the object. When you activate this option, the dialog displays a message indicating that the line update mode is now active.

Row update processing is currently limited to certain policy members and certain fields within those policy members.

Online Help

The MAINVIEW Batch Optimizer user interface provides explanations of all panels, fields, and messages in the ISPF interface and dialogs. Panel-level help displays a pop-up window that explains the purpose of the panel, provides information about using the panel, and provides a description of all fields on the panel. Field-level help displays a pop-up window that describes the purpose of the field and possible values that you can enter in the field. Message-level help displays a pop-up window that describes an error message, the reason for the error message, and possible responses to the error. Reference phrase help displays a pop-up window that describes the highlighted word or phrase. These highlighted phrases typically appear on help panels. Help is also available for the action bar.

You can activate the help feature by, pressing **F1** (HELP), selecting the Help option from the Help pull-down menu, or typing **HELP** on the command line and pressing **Enter**. The type of help displayed when you activate the help feature depends on the cursor’s position.

Panel Help

The dialog displays the panel-level help when you activate the help feature with the cursor on the command line or on any part of the panel other than the action bar or a choice entry or data entry field.

Field Help

The dialog displays the field-level help when you activate the help feature with the cursor on a choice entry field, a data entry field, the heading of a data entry field, or some output-only fields.

Message Help

The dialog displays the message-level help when you activate the help feature after the dialog displays a "long message" pop-up on the screen. The message help clarifies the error causing the message, describes the reason for the error message, and suggests actions to resolve the error.

Reference Phrase Help

The dialog displays reference phrase help when you position the cursor on a highlighted field within a help panel and activate the help feature. This facility offers further explanations of phrases or words mentioned in an already displayed panel, field, or message help panel.

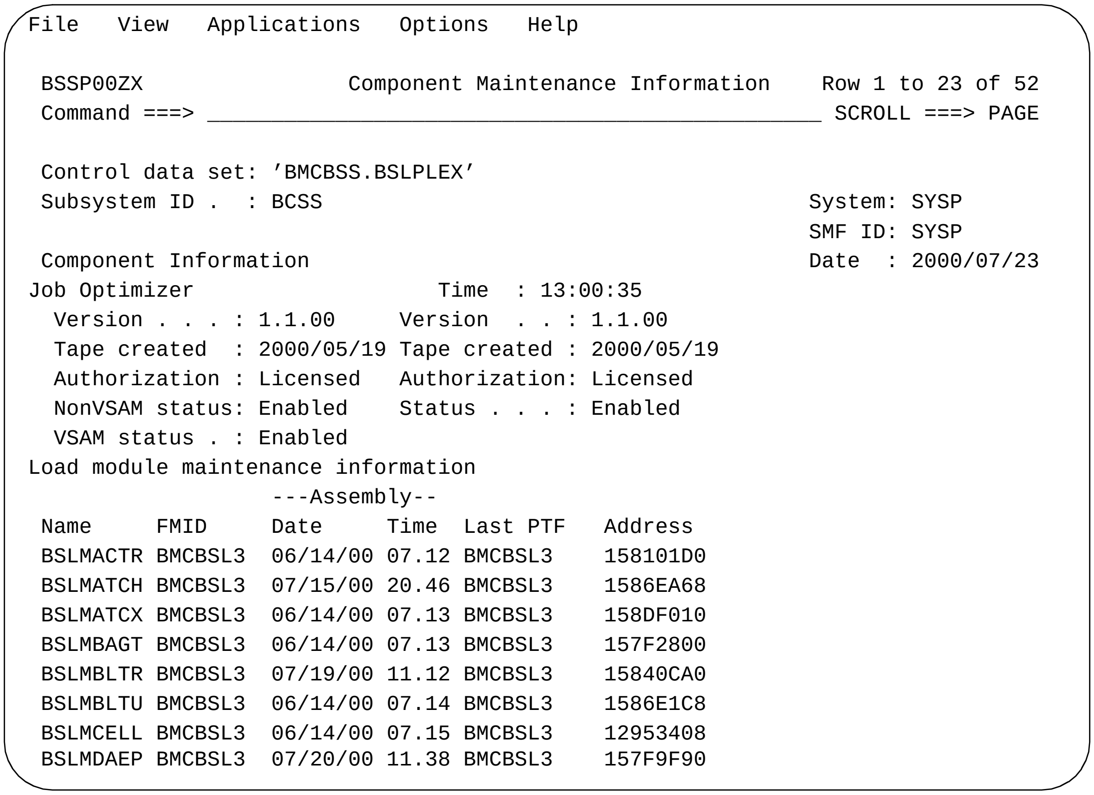
Other User Interface Tasks

Besides editing the control data set members, the dialog provides access to other facilities of MAINVIEW Batch Optimizer. When contacting BMC Software Product Support, you will need to obtain the product version information. You can obtain this as well as module maintenance and environmental version information through the dialog. Alternatively, you can install BMC Software product authorization codes through the dialog.

Product Version and Maintenance Level Information

You can access the product version and maintenance level information by using the **MN** line command on the MAINVIEW Batch Optimizer Objects List panel to select the active BatchPlex member (BPLEXnn). Figure B-16 shows the Components Maintenance Information panel.

Figure B-16 Components Maintenance Information Panel



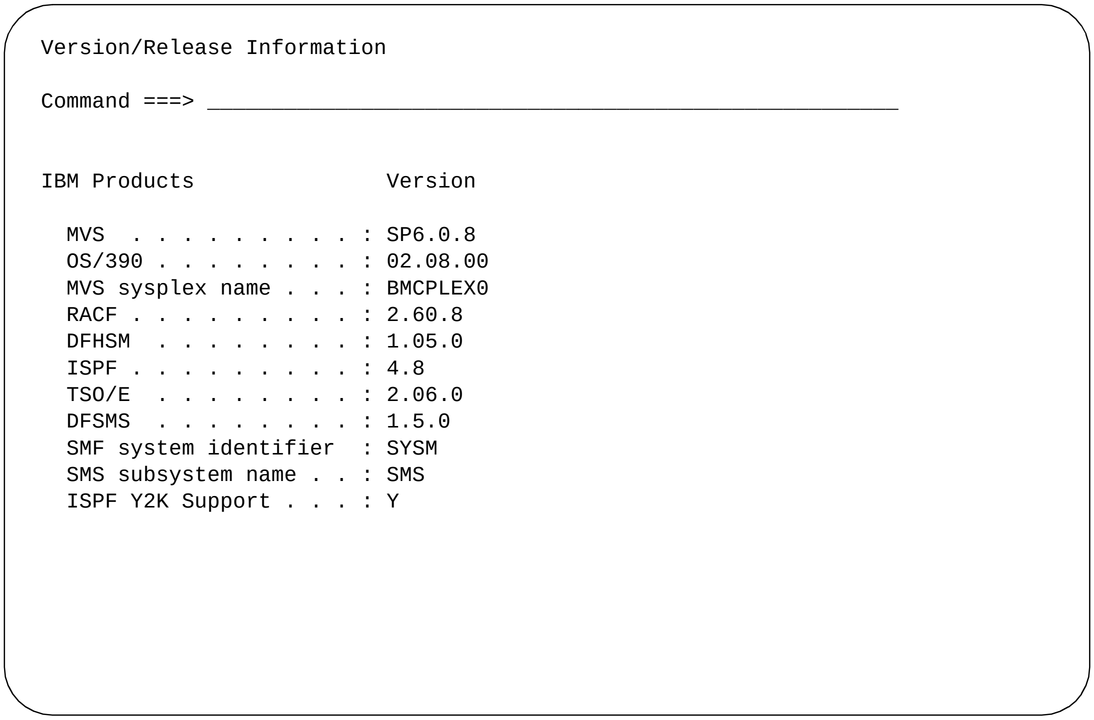
```
File   View   Applications   Options   Help

 BSSP00ZX                Component Maintenance Information    Row 1 to 23 of 52
 Command ===> ________________________________________________ SCROLL ===> PAGE

 Control data set: ’BMCBSS.BSLPLEX’
 Subsystem ID .  : BCSS                                      System: SYSP
                                                             SMF ID: SYSP
 Component Information                                       Date  : 2000/07/23
Job Optimizer                   Time  : 13:00:35
  Version . . . : 1.1.00     Version  . . : 1.1.00
  Tape created  : 2000/05/19 Tape created : 2000/05/19
  Authorization : Licensed   Authorization: Licensed
  NonVSAM status: Enabled    Status . . . : Enabled
  VSAM status . : Enabled
Load module maintenance information
                   ---Assembly--
 Name     FMID     Date     Time  Last PTF   Address
 BSLMACTR BMCBSL3  06/14/00 07.12 BMCBSL3    158101D0
 BSLMATCH BMCBSL3  07/15/00 20.46 BMCBSL3    1586EA68
 BSLMATCX BMCBSL3  06/14/00 07.13 BMCBSL3    158DF010
 BSLMBAGT BMCBSL3  06/14/00 07.13 BMCBSL3    157F2800
 BSLMBLTR BMCBSL3  07/19/00 11.12 BMCBSL3    15840CA0
 BSLMBLTU BMCBSL3  06/14/00 07.14 BMCBSL3    1586E1C8
 BSLMCELL BMCBSL3  06/14/00 07.15 BMCBSL3    12953408
 BSLMDAEP BMCBSL3  07/20/00 11.38 BMCBSL3    157F9F90
```

BMC Software Product Support may ask you to refer to this information when attempting to diagnose a reported problem.

Environment Version and Release Information

You can access the version and release information by selecting the Version information pull-down choice of the Options action bar item or by entering the command **VERSION** on the command line. Figure B-17 shows the Version/Release Information panel.

Figure B-17 Version/Release Information Panel



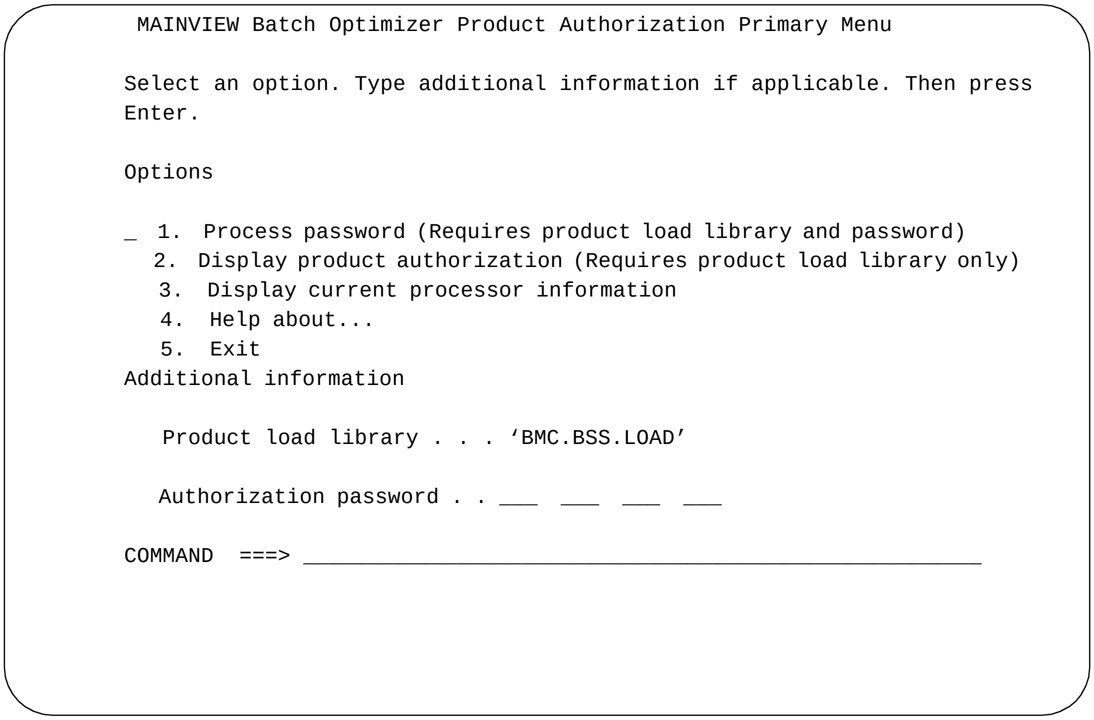
```
Version/Release Information

Command ===> ______________________________________________________


IBM Products               Version

  MVS  . . . . . . . . . . : SP6.0.8
  OS/390 . . . . . . . . . : 02.08.00
  MVS sysplex name . . . : BMCPLEX0
  RACF . . . . . . . . . . : 2.60.8
  DFHSM  . . . . . . . . . : 1.05.0
  ISPF . . . . . . . . . . : 4.8
  TSO/E  . . . . . . . . . : 2.06.0
  DFSMS  . . . . . . . . . : 1.5.0
  SMF system identifier  : SYSM
  SMS subsystem name . . : SMS
  ISPF Y2K Support . . . : Y
```

It is useful to have the information displayed by this panel on hand when contacting BMC Software Product Support.

BMC Software Product Authorization User Interface

You can access the product authorization facility by selecting the BMC Software Security Facility pull-down choice of the Applications action bar item or entering the command **SECURITY** on the command line. Figure B-18 shows the Product Authorization Primary Menu panel.

Figure B-18 Product Authorization Primary Menu Panel



```
 MAINVIEW Batch Optimizer Product Authorization Primary Menu

Select an option. Type additional information if applicable. Then press
Enter.

Options

_  1.  Process password (Requires product load library and password)
   2.  Display product authorization (Requires product load library only)
   3.  Display current processor information
   4.  Help about...
   5.  Exit
Additional information

   Product load library . . . 'BMC.BSS.LOAD'

   Authorization password . . ___  ___  ___  ___

COMMAND  ===> _____________________________________________________
```

In this facility, you can add new licensing information, display existing licensing information, or display information about the current processor. It is on this panel that you will include any licensing passwords provided to you by BMC Software.

Appendix C User Control Facility Internal Policy

This appendix describes the User Control Facility (UCF) that is used in Job Optimizer. This appendix discusses the following topic:

BMC Software, Inc., Confidential and Proprietary Information

Internal UCF Table Definitions

User Control Facility (UCF) provides you with an additional means of controlling job performance processing. BMC Software has included UCF entries for certain IBM and vendor-supplied programs. Table C-1 lists these entries.

Table C-1 Internal UCF Table Definitions (Part 1 of 2)

Name	Description	Type of Action Performed
ADRDSSU	The main entry point module of IBM's DFDSS DASD backup and restore utility.	SERIALIZE
ADRDDDS	A module of IBM's DFDSS DASD backup and restore utility.	SERIALIZE
ADRDFRAG	A module of IBM's DFDSS DASD backup and restore utility.	SERIALIZE
ADREFRAG	A module of IBM's DFDSS DASD backup and restore utility.	SERIALIZE
ADRLDFRG	A module of IBM's DFDSS DASD backup and restore utility.	SERIALIZE
ADRDTDSC	A module of IBM's DFDSS DASD backup and restore utility.	SERIALIZE
ADRDTDS	A module of IBM's DFDSS DASD backup and restore utility.	SERIALIZE
ADRTDDSC	A module of IBM's DFDSS DASD backup and restore utility.	SERIALIZE
ADRTDDS	A module of IBM's DFDSS DASD backup and restore utility.	SERIALIZE
ADSMI002	Sterling Software's DMS DASD space management utility.	BYPASS JOB
ARCBDSN	A module of IBM's DF/HSM DASD Space management utility.	SERIALIZE
ARCBUDS	A module of IBM's DF/HSM DASD Space management utility.	SERIALIZE
ARCMDSN	A module of IBM's DF/HSM DASD Space management utility.	SERIALIZE
ARCMDSUV	A module of IBM's DF/HSM DASD Space management utility.	SERIALIZE
ARCRSTR	A module of IBM's DF/HSM DASD Space management utility.	SERIALIZE
DFHSIP	A module of CICS.	BYPASS JOB
FDR	A module of Innovation Data Processing's data backup utilities.	BYPASS JOB
FDRTCOPY	A module of Innovation Data Processing's data backup utilities.	BYPASS JOB
FDRTSEL	A module of Innovation Data Processing's data backup utilities.	BYPASS JOB
FDRQUERY	A module of Innovation Data Processing's data backup utilities.	BYPASS JOB
FDRDSF	A module of Innovation Data Processing's data backup utilities.	BYPASS JOB
FDRCOPY	A module of Innovation Data Processing's data backup utilities.	BYPASS JOB
FDRREORG	A module of Innovation Data Processing's data backup utilities.	BYPASS JOB
FDRREOZO	A module of Innovation Data Processing's data backup utilities.	BYPASS JOB
FDRSARLR	A module of Innovation Data Processing's data backup utilities.	BYPASS JOB
HEWL	Linkage Editor.	SERIALIZE

BMC Software, Inc., Confidential and Proprietary Information

Table C-1 Internal UCF Table Definitions (Part 2 of 2)

Name	Description	Type of Action Performed
IDCAMS	VSAM utility.	SERIALIZE
IEWL	Linkage Editor.	SERIALIZE
U11RMS	Computer Associates' CA-11 product.	RESTART
IKJEFT01	A module of TSO's batch mode.	SERIALIZE
IKJEFT1A	A module of TSO's batch mode.	SERIALIZE
IKJEFT1B	A module of TSO's batch mode.	SERIALIZE
SASLPA	SAS Institute, Inc.	SERIALIZE
SASHOST	SAS Institute, Inc.	SERIALIZE
SASSBSTR	SAS Institute, Inc.	SERIALIZE

BMC Software, Inc., Confidential and Proprietary Information

Appendix D Job Optimizer for DB2 and SQL Statements

This appendix explains what serialization of job steps means and what data constraints are in Job Optimizer for DB2. It also explains what SQL statements will cause serialization or data constraints. This appendix also discusses support of referential integrity in Job Optimizer for DB2. This appendix discusses the following topics:

Serialization of Job Steps

Serialization of a job step means that the job step must run alone. Any job steps prior to the serialized job step *must complete before* that job step begins. The serialized job step then runs alone. After the serialized job step completes, any subsequent job steps may begin.

Any job step that contains one of the following SQL statements will serialize:

- ALTER DATABASE
- ALTER INDEX
- ALTER STOGROUP
- ALTER TABLE
- ALTER TABLESPACE
- CONNECT
- CREATE ALIAS
- CREATE DATABASE
- CREATE GLOBAL TEMPORARY TABLE
- CREATE INDEX
- CREATE STOGROUP
- CREATE SYNONYM
- CREATE TABLE
- CREATE TABLE SPACE
- CREATE VIEW
- DROP ALIAS
- DROP DATABASE
- DROP INDEX
- DROP PACKAGE
- DROP PROGRAM
- DROP STOGROUP
- DROP SYNONYM
- DROP TABLE
- DROP TABLESPACE
- DROP VIEW
- EXECUTE
- GRANT
- PREPARE
- RELEASE
- RENAME
- REVOKE

BMC Software, Inc., Confidential and Proprietary Information

Data Constraints Caused by SQL Statements

A data constraint can occur between two or more steps in the same job. This happens when a SQL statement exists in two or more job steps of the same job that updates the job against the same DB2 table. A data constraint also can occur if a job step contains a SQL statement that updates a DB2 table and subsequent job steps read that DB2 table. Job steps that have data constraints can be processed in parallel with any *other* job steps with which they have no data constraints. Any job step that contains one of the following SQL statements can encounter data constraints with other job steps:

- DELETE FROM
- INSERT INTO
- LOCK TABLE (only in EXCLUSIVE mode, not SHARE mode)
- UPDATE

Support of Referential Integrity

If the DELETE SQL statement is issued for a DB2 table involved in referential integrity relationships, Job Optimizer for DB2 will find all the dependent tables for that parent table and process them in the following manner:

- For dependent tables of the DELETE table that have a CASCADE DELETERULE, Job Optimizer for DB2 places a data constraint on these tables.

Job Optimizer for DB2 checks the DELETERULES of the dependent tables of these dependent tables. Job Optimizer for DB2 continues to track these dependent tables and their dependent tables as updated tables as long as their DELETERULES are CASCADE. If Job Optimizer for DB2 encounters a CASCADE dependent table's dependent table with a different DELETERULE, Job Optimizer for DB2 will perform the logic for that relationship (See the following bulleted statements).

- For dependent tables of the DELETE table that have a DELETERULE of SET NULLS, Job Optimizer for DB2 tracks these dependent tables as updated tables.

- For dependent tables of the DELETE table that have a DELETERULE of RESTRICT or NO ACTION, Job Optimizer for DB2 tracks these dependent tables as read-only tables.

If an INSERT or UPDATE SQL statement is issued for a DB2 table involved in referential integrity relationships, Job Optimizer for DB2 will find all the parent tables for that dependent table and process them in the following manner:

- For parent tables of the INSERT or UPDATE table that have a CASCADE DELETERULE, Job Optimizer for DB2 tracks these parent tables as updated tables.

Job Optimizer for DB2 checks the DELETERULES of the parent tables of these parent tables. Job Optimizer for DB2 continues to track these parent tables and their parent tables as updated tables as long as their DELETERULES are CASCADE. If Job Optimizer for DB2 encounters a CASCADE parent table’s parent table with a different DELETERULE, Job Optimizer for DB2 will perform the logic for that relationship (see the following bulleted statements).

- For parent tables of the INSERT or UPDATE table that have a DELETERULE of SET NULLS, Job Optimizer for DB2 tracks these parent tables as updated tables.

- For parent tables of the INSERT or UPDATE table that have a DELETERULE of RESTRICT or NO ACTION, Job Optimizer for DB2 tracks these parent tables as read-only tables.

Appendix E Migrating from BatchPipes to Job Optimizer Pipes

This appendix explains Job Optimizer Pipes support for IBM BatchPipes users and how to convert from BatchPipes to Job Optimizer Pipes. This appendix discusses the following topics:

BMC Software, Inc., Confidential and Proprietary Information

Overview

BatchPipes users can use Job Optimizer Pipes in one of the following methods:

- Migrate to Job Optimizer Pipes with minimal changes to their environment
- Convert their BatchPipes pipes to Job Optimizer Pipes
- A combination of the above methods.

BatchPipes to Job Optimizer Pipes migration can be done smoothly without changing the JCL of the jobs. Job Optimizer Pipes supports BatchPipes SUBSYS= subsystem parameters. When migrating from BatchPipes the jobs' JCL can remain untouched. Job Optimizer Pipes will analyze the BatchPipes subsystem parameters and will define and manage pipes accordingly.
The following list provides the IBM SmartBatch for OS/390 and BatchPipes releases that are supported by Job Optimizer Pipes:

- IBM SmartBatch for OS/390 v1.2
- BatchPipes v2.1

BatchPipes to Job Optimizer Pipes conversion means using Job Optimizer Pipes pipe rules to define and manage pipes instead of BatchPipes subsystem parameters. The conversion can be done optionally, depending on your requirements. Rules can be defined for new jobs that have to use pipes, and existing jobs can be changed to use rules instead of BatchPipes subsystem parameters.

BatchPipes to Job Optimizer Pipes Migration

When migrating from BatchPipes, Job Optimizer Pipes will analyze the BatchPipes subsystem parameters and will define and manage pipes accordingly. BatchPipes pipes migrated to Job Optimizer Pipes are called “BatchPipes-Compatible Pipes.”

This section describes Job Optimizer Pipes support for BatchPipes-Compatible Pipes. It describes the migration steps, how Job Optimizer Pipes processes BatchPipes subsystem parameters and general differences between BatchPipes processing and Job Optimizer Pipes processing of BatchPipes subsystem parameters.

Migration Steps

Migrating from BatchPipes to Job Optimizer Pipes can be performed in two steps:

Step 1–Phased Migration

Phased migration means moving selected jobs or applications from using BatchPipes to using Job Optimizer Pipes. Job Optimizer Pipes provides the ability to dynamically redirect jobs using BatchPipes (via the SUBSYS= JCL parameter) to use Job Optimizer Pipes, although the JCL still contains the BatchPipes subsystem name.

In order to perform phased migration the following must be defined:

- Define the names of all the BatchPipes subsystems to Job Optimizer Pipes. This is performed via initialization parameter “BatchPipes subsystem IDs” (see Chapter 10, “Defining Job Optimizer Pipes Initialization Parameters,” for more information).

- Define Pipe Policy Migration Rules. These are standard Pipe rules, where the BatchPipes migration option field is set to Y (see Figure 9-10 on page 9-24). A Pipe Policy Migration Rule can be set for a specific dataset or a set of datasets (using a mask for the Pipe data set name field). It can be set for a specific job or a set of jobs (by setting the reader/writer selection parameters to be specific names or masks). When set for specific jobs, the pipe dataset name can be set to * to allow all BatchPipes pipes accessed by the jobs to be migrated to Job Optimizer Pipes. The other fields of the pipe rule are ignored as the information is taken from the BatchPipes subsystem parameters. More than one Pipe Policy Migration Rule can be defined, according to the selection criteria of the pipes and jobs candidate for phased migration. After the Pipe Policy Migration Rules are defined, they should be activated.

- When jobs candidate for BatchPipes phased migration are submitted, BatchPipes subsystem should be active. The migration process starts when the job starts execution. Process required from BatchPipes before this stage (for example, during the jobs’ JCL conversion) is still performed by BatchPipes. When the job starts execution, Job Optimizer Pipes will intercept the pipes candidate for phased migration.

Step 2–Full Migration

After migrating selected jobs and applications, full migration to Job Optimizer Pipes should be performed. Full migration means replacing BatchPipes with Job Optimizer Pipes for all jobs. The full migration involves the following steps:

- Stop BatchPipes address space.
- Stop Job Optimizer Pipes address space.
- Remove the BatchPipes subsystem name from the BatchPipes subsystem IDs in Job Optimizer Pipes initialization parameter. See Chapter 10, “Defining Job Optimizer Pipes Initialization Parameters,” for more information.
- Define the Job Optimizer Pipes subsystem name (parameter Pipes subsystem ID in the BatchPlex Definition panel)) to be the same as the BatchPipes subsystem.
- If Pipe Policy Migration Rules were defined during Phased Migration, remove them from the Pipe Policy List (member PIPLSTxx) pointed from BatchPlex parameters.
- Start Job Optimizer Pipes in all required systems.
- Jobs can now be submitted to use Job Optimizer Pipes instead of BatchPipes.

BatchPipes-Compatible Pipes Support

A pipe is processed in BatchPipes compatible mode when SUBSYS= parameter is specified with BatchPipes subsystem parameters or with no subsystem parameters. When a pipe is processed in BatchPipes-Compatible mode, the message BMC282348I is issued, and PIPE-FLG=BATCH-PI is displayed in the message BMC282900I.

Job Optimizer Pipes defines and manages a BatchPipes-Compatible pipe using the BatchPipes subsystem parameters specified in JCL, the DCB parameters specified in JCL and default values specified in Pipe Defaults Rule JOPDEFRL.

- Table E-1 on page E-6 describes how BatchPipes subsystem parameters are supported by Job Optimizer Pipes.

- Table E-2 on page E-13 describes which DCB parameters are supported by Job Optimizer Pipes for BatchPipes-Compatible Pipes.

- JOPDEFRL defines default values for various pipe parameters. Only specific parameters are used for BatchPipes-Compatible pipes.These parameters should be set with values according to the values set in BatchPipes initialization parameters member ASFPBPxx. See “Editing or Viewing Pipe Rule Defaults” on page 9-18 for explanations how to set the pipe default values. Table E-3 on page E-14 details the BatchPipes initialization parameters that are supported and their corresponding Job Optimizer Pipes default options.

There are several major differences between BatchPipes and Job Optimizer Pipes regarding pipe processing:

- In BatchPipes, any number of participants can join an active pipe. Parameters ALLOCSYNC and OPENSYNC define how many participants are required at each point but more participants can join before or after these synchronization points. In Job Optimizer Pipes the number of participants allowed to join the pipe is controlled by parameters ALLOCSYNC and OPENSYNC (which are also used to define how many participants are required at each synchronization point) and by initialization parameters ADD-RDRS and ADD-WTRS:

  — When ADD-RDRS=NO and ADD-WTRS=NO are specified (ADD-RDR=0 and ADD-WTR=0 are displayed in pipe parameter message BMC282900I), only the number of participants specified in ALLOCSYNC/OPENSYNC are allowed (and required) to join the pipe. If ALLOCSYNC/OPENSYNC parameters are not specified, 1 reader and 1 writer are allowed.

  — When ADD-RDRS=YES is specified (ADD-RDR=99 is displayed in pipe parameter message BMC282900I), the number of participants specified in ALLOCSYNC/OPENSYNC is the minimum number of participants allowed (and required) to join the pipe. Any number of readers are allowed to join the pipe as long as the pipe is active.

- — When ADD-WTRS=YES is specified (ADD-WTR=99 is displayed in pipe parameter message BMC282900I), the number of participants specified in ALLOCSYNC/OPENSYNC is the minimum number of participants allowed (and required) to join the pipe. Any number of writers are allowed to join the pipe as long as the pipe is active.

Warning! Allowing additional readers/writers to join the pipe will cause significant performance degradation. Therefore, both ADD-RDRS and ADD-WTRS parameters should be set to NO unless absolutely necessary. If multiple writers/readers are required for specific pipes, their subsystem parameters should be adjusted (ALLOCSYNC, ALLOCNOW, OPENSYNC, OPENNOW), or the pipe can be converted to be a Job Optimizer Pipes pipe (using a rule).

- In BatchPipes, a reader can close and re-open the pipe several times during its process. In Job Optimizer Pipes, a reader can re-open the pipe only if it did not read anything from the pipe. If the reader opens the pipe, reads data, closes and opens again, it will fail.

- In BatchPipes, a participant is allowed to close the pipe and re-open as a different participant type (for example a writer can write to the pipe, close, and open as a reader). This is not allowed in Job Optimizer Pipes. If a participant wants to connect to the pipe with a different type, it has to deallocate the pipe and allocate again.

Table E-1 BatchPipes Subsystem Parameters Supported by Job Optimizer Pipes (Part 1 of 7)

BatchPipes Parameter	Job Optimizer Pipes Field Name in Message BMC282900I	Default	Description
ALLOCNOW	SYNC-TYP=ALLOC+ WTR/RDR-FLAG=SKP-ALCS	—	Allows a job that might otherwise wait during allocation synchronization processing to complete allocation. The job can request synchronization at allocation and count towards the number of participants required for synchronization, but the job itself does not wait during allocation.

ALLOCSYNC=n[1]	SYNC-TYP=ALLOC+
	REQ-PRTS=n
		—	Synchronize the pipe allocation process. Pipe allocation is held up until a minimum number of participants have allocated the pipe. REQ-PRTS specifies the number of participants that must allocate the pipe before processing continues.

Table E-1 BatchPipes Subsystem Parameters Supported by Job Optimizer Pipes (Part 2 of 7)

BatchPipes Parameter	Job Optimizer Pipes Field Name in Message BMC282900I	Default	Description
CLOSESYNC	SYNC-TYP=CLOSE	—	Synchronize pipe close process. When a participant has closed a pipe, its processing is held up until all participants have closed the pipe.
EOFREQUIRED=NO[4]	RDR-FLAG=NO-EOFWT	Value in JOPDEFRL (Default Rule)	The reader can close the pipe without receiving an EOF indication.
EOFREQUIRED=YES[4]	RDR-FLAG=WAIT4EOF	Value in JOPDEFRL (Default Rule)	The reader should not close the pipe until an EOF indication is received. Otherwise, the reader abends.
ERC=CONT	—	ERC=CONT	Indicates that on early reader close, if additional readers are allowed to join the pipe, the writer will continue writing to the pipe. Otherwise, the writer will abend.
ERC=DUMMY	PIPE-FLG=ERC=DUMY	ERC=CONT	Indicates that on early reader close the writer will stop writing to the pipe even if more readers can join. If fitting was specified, the writer will continue writing to the fitting. If fitting was not specified, the writer will continue its process without writing anywhere.

ERRPROP=CANCEL[3,5]	RD-EROPT=IMMED
	WT-EROPT=IMMED
		ERRPROP=ABEND	Fail the participant when another participant is in error or when there is an error in the pipe process.
			For reader, when an error occurs, the reader either receives an I/O error or abends, depending on its processing stage.
			For writer, when an error occurs, the writer either receives an I/O error or abends, depending on its processing stage.

ERRPROP=ABEND[3, 5]	RD-EROPT=IMMED
	WT-EROPT=IMMED
		ERRPROP=ABEND	Fail the participant when another participant is in error or when there is an error in the pipe process.
			For reader, when an error occurs the reader either receives an I/O error or abends, depending on its processing stage.
			For writer, when an error occurs the writer either receives an I/O error or abends, depending on its processing stage.

Table E-1 BatchPipes Subsystem Parameters Supported by Job Optimizer Pipes (Part 3 of 7)

BatchPipes Parameter	Job Optimizer Pipes Field Name in Message BMC282900I	Default	Description

ERRPROP=CONT[3, 5]	RD-EROPT=IGNORE
	WT-EROPT=IGNORE
		ERRPROP=ABEND	Ignore other participant failure.
			For reader, when an error occurs, the reader continues to access the pipe. When all data has been read from the pipe, the reader ends normally.
			For writer, when an error occurs, the writer continues writing to the pipe as long as there are readers connected to the pipe. If there are no readers connected to the pipe, it will continue its process without writing data to the pipe. If fitting is specified, the writer will continue writing data to the fitting. If fitting was not specified, the writer will continue its process without writing anywhere. When the process finishes, the writer ends normally.

ERRPROP=DUMMY[3, 5]	RD-EROPT=IMMED
	WT-EROPT=IGNORE
		ERRPROP=ABEND	Stop accessing the pipe when another participant fails or when there is an error in the pipe process.
			For reader, when an error occurs, the reader either receives an I/O error or abends, depending on its processing stage.
			For writer, when an error occurs, the writer continues writing to the pipe as long as there are readers connected to the pipe. If there are no readers connected to the pipe, it will continue its process without writing data to the pipe. If fitting is specified, the writer will continue writing data to the fitting. If fitting was not specified, the writer will continue its process without writing anywhere. When the process finishes, the writer ends normally.

FIT=fitting specification	PIPE-FLG=FITTING+
	DD-NUM=n+
	FILE-DDn=file-ddnames
		—	Identifies the fitting specification. Fitting is supported for hardening the pipe data only (BPCOPY/BPREAD/QSAM/BPWRITE commands). Hardening can be done to 1-5 files (up to 5 ddnames may appear in BPCOPY/QSAM commands). If other BatchPipesWorks commands are specified or if hardening is done to more than 5 files, the fitting is not supported and the job fails.

Table E-1 BatchPipes Subsystem Parameters Supported by Job Optimizer Pipes (Part 4 of 7)

BatchPipes Parameter	Job Optimizer Pipes Field Name in Message BMC282900I	Default	Description

FITDD=ddname
CMT=comment delimiter
	PIPE-FLG=FITTING+
	FITDD=ddname+
	DD-NUM=n+
	FILE-DDn=file-ddnames
		—	Identifies the DD statement pointing to the fitting specification and the comment character used inside the fitting specification. See FIT= above for the fitting specification syntax rules.

IDLE=nnnn[3, 4]	NOOP-WT=nnnn	value in JOPDEFRL (Default Rule)	Time interval (in minutes) during which a participant must access the pipe. If the participant does not access the pipe within the specified time interval, a message which requires a reply is sent to the operator console asking if the participant should abend or should wait for an additional time interval.
IDLE=OFF [3, 4]	NOOP-WT=9999	value in JOPDEFRL (Default Rule)	
NOEOF	WTR-FLAG=NOEOF	—	The writer closes and deallocates a pipe without sending an EOF signal to the pipe. If not specified, the writer sends an EOF signal when closing the pipe (WTR-FLAG= EOF-CLOSE is displayed in message BMC282900I)
OPENNOW	SYNC-TYP=OPEN+ WTR/RDR-FLAG= SKP-OPNS	—	Allows a job that might otherwise wait during open synchronization processing to complete open. The job can request synchronization at open and count towards the number of readers/writers required for synchronization, but the job itself does not wait during open.

OPENSYNC= (R=x, W=y) [1]	SYNC-TYP=OPEN+
	REQ-RDR=x+
	REQ-WTR=y
		OPENSYNC= (R=1, W=1)	Synchronize pipe open process. Data transfer is held up until a minimum number of participants have opened the pipe.
			REQ-RDR specifies the minimum number of readers that must open the pipe before data transfer begins.
			REQ-WTR specifies the minimum number of writers that must open the pipe before data transfer begins.

PIPEDEPTH=nnn[3, 4, 5]	BUFF#=nnnn	value in JOPDEFRL (Default Rule)	Number of buffers to allocate for the pipe. There is no maximum buffer number value (MAXBUFNO) in Job Optimizer Pipes.

Table E-1 BatchPipes Subsystem Parameters Supported by Job Optimizer Pipes (Part 5 of 7)

BatchPipes Parameter	Job Optimizer Pipes Field Name in Message BMC282900I	Default	Description
TERMSYNC	SYNC-TYP=DEALLOC+ ERROR-CC=0	—	Synchronizes pipe deallocation process. When the deallocation is done as part of step termination, the deallocation of all pipes of the step having TERMSYNC is synchronized with all the participants of these pipes. The step condition code is compared against the 'cc' parameter of each pipe. If the step abended or the condition code is equal to or higher than the 'cc' parameter of any pipe, the step is abended and the error is distributed to all the participants of all the pipes causing them to abend. Otherwise, Job Optimizer Pipes waits for all the participants of all the pipes of this step having TERMSYNC to arrive at deallocation. When all the participants arrive successfully at deallocation, the deallocation is completed and the step terminates. When the deallocation is done as a result of dynamic deallocation (either explicitly or as a result of FREE=CLOSE specification), Job Optimizer Pipes synchronizes the deallocation of the specific pipe. 'cc' is ignored as it is not known at this stage. A pipe is not eligible for synchronization when there was a previous error on the pipe or when Early Readers Close occurs.
TERMSYNC=cc	SYNC-TYP=DEALLOC+ ERROR-CC=cc	—	
WAIT=nnnn[2, 3, 4]	I/O-WT=nnnn	value in JOPDEFRL (Default Rule)	Maximum time (in minutes) to wait for data to be transferred to or from the pipe. If a data transfer does not occur within the specified time limit, a message which requires a reply is sent to the operator console asking whether the application should abend or should wait for an additional time interval.
WAIT=OFF[2, 3, 4]	I/O-WT=9999	value in JOPDEFRL (Default Rule)	

Table E-1 BatchPipes Subsystem Parameters Supported by Job Optimizer Pipes (Part 6 of 7)

BatchPipes Parameter	Job Optimizer Pipes Field Name in Message BMC282900I	Default	Description
WAITALLOC=nnnn [2, 3, 4]	SYNC-WT=nnnn	value in JOPDEFRL (Default Rule	Maximum time (in minutes) that a participant waits for each synchronization point to be reached. If synchronization requirements are not satisfied within the specified time interval, a message which requires a reply is sent to the operator console asking whether the application should fail or should wait for an additional time interval.
WAITALLOC=OFF[2, 3, 4]	SYNC-WT=9999	value in JOPDEFRL (Default Rule	
WAITCLOSE=nnnn[2, 3, 4]	SYNC-WT=nnnn	value in JOPDEFRL (Default Rule)	Maximum time (in minutes) that a participant waits for each synchronization point to be reached. If synchronization requirements are not satisfied within the specified time interval, a message which requires a reply is sent to the operator console asking whether the application should fail or should wait for an additional time interval.
WAITCLOSE=OFF[2, 3, 4]	SYNC-WT=9999	value in JOPDEFRL (Default Rule)	
WAITEOF=nnnn[2, 3, 4]	I/O-WT=nnnn	value in JOPDEFRL (Default Rule)	Maximum time (in minutes) to wait for data to be transferred to or from the pipe. If a data transfer does not occur within the specified time limit, a message which requires a reply is sent to the operator console asking whether the application should abend or should wait for an additional time interval.
WAITEOF=OFF[2, 3, 4]	I/O-WT=9999	value in JOPDEFRL (Default Rule)	
WAITOPEN=nnnn[2, 3, 4]	SYNC-WT=nnnn	value in JOPDEFRL (Default Rule)	Maximum time (in minutes) that a participant waits for each synchronization point to be reached. If synchronization requirements are not satisfied within the specified time interval, a message which requires a reply is sent to the operator console asking whether the application should fail or should wait for an additional time interval.
WAITOPEN=OFF[2, 3, 4]	SYNC-WT=9999	value in JOPDEFRL (Default Rule)	

Table E-1 BatchPipes Subsystem Parameters Supported by Job Optimizer Pipes (Part 7 of 7)

BatchPipes Parameter	Job Optimizer Pipes Field Name in Message BMC282900I	Default	Description
WAITTERM=nnnn[2, 3, 4]	SYNC-WT=nnnn	value in JOPDEFRL (Default Rule)	Maximum time (in minutes) that a participant waits for each synchronization point to be reached. If synchronization requirements are not satisfied within the specified time interval, a message which requires a reply is sent to the operator console asking whether the application should fail or should wait for an additional time interval.
WAITTERM=OFF[2, 3, 4]	SYNC-WT=9999	value in JOPDEFRL (Default Rule)	

[1] Parameters ALLOCSYNC and OPENSYNC can also impact the number of participants allowed to join the pipe. See “BatchPipes-Compatible Pipes Support” above for a detailed explanation.

[2] In several cases, different BatchPipes subsystem parameters are given the same Job Optimizer Pipes meaning. If multiple such parameters are specified for a participant, the value of the last is used as the Job Optimizer Pipes value. These parameters are:

- WAITALLOC, WAITOPEN, WAITCLOSE, WAITTERM are translated to Job Optimizer Pipes Sync-wait-time option (displayed as SYNC-WT in pipe parameters message BMC282900I)
- WAIT, WAITEOF are translated to Job Optimizer Pipes IO-wait-time option (displayed as I/O-WT in pipe parameters message BMC282900I)

[3] In BatchPipes, most subsystem parameters are participant-related - the values come from the subsystem parameters specified in each participants' JCL and are used for this participant processing only. In Job Optimizer Pipes, most parameters are pipe-related - the values are specified in the rule and are used for processing all the participants matching the rule. For BatchPipes-Compatible pipes, the following participant-related parameters are used as pipe-related parameters:

- Sync-wait-time, I/O-wait-time, IDLE-time and PIPEDEPTH - the first participant sets the values for these parameters. These values will be used by all participants of the pipe. A warning message is issued to all participants with values different than the pipe.
- ERRPROP - the first participant from each type (reader/writer) sets the error option value for all the participants from the same type. A warning message is issued to all participants with values different than the pipe.

[4] Setting JOPDEFRL default values is described under “BatchPipes-Compatible Pipes Support” on page E-4.

[5] These parameters have corresponding DCB parameters. If a parameter is not specified, the value from its corresponding DCB parameter is used, if specified. If the DCB parameter is not specified, the default is used. DCB parameters support for BatchPipes-Compatible Pipes is detailed in Table E-2 on page E-13.

Table E-2 DCB parameters supported by Job Optimizer Pipes

DCB Parameter	Corresponding BatchPipes Subsystem Parameter	Job Optimizer Pipes Field Name in Message BMC282900I	Description
LRECL, RECFM, BLKSIZE	—	LRECL, RECFM, BLKSIZE	Pipe attributes. If specified in JCL, the attributes must be the same in all the participants. If not specified, the pipe attributes will be set using the attributes specified in the application. If block size is not specified, default block size is set. For fixed blocked records the default block size will be the largest multiple of the LRECL that is less or equal 32760. For variable blocked records the default block size will be 32760.
BUFNO	PIPEDEPTH	BUFF#	Number of buffers to allocate for the pipe. This parameter is used if PIPEDEPTH is not specified. If value less than 3 is specified, 3 will be used as the number of buffer for the pipe.

EROPT=ACC	ERRPROP=CONT	RD-EROPT=IGNORE or
		WT-EROPT=IGNORE
			See the description of ERRPROP=CONT in Table E-1 on page E-6.

EROPT=SKP	ERRPROP=DUMMY	RD-EROPT=IMMED or
		WT-EROPT=IGNORE
			See the description of ERRPROP=DUMMY in Table E-1 on page E-6.

EROPT=ABE	ERRPROP=ABEND	RD-EROPT=IMMED or
		WT-EROPT=IMMED
			See the description of ERRPROP=ABEND in Table E-1 on page E-6.

Table E-3 shows the BatchPipes initialization parameters and JOPDEFRL fields.

Table E-3 BatchPipes Initialization Parameters and JOPDEFRL Fields

BatchPipes parameter	JOPDEFRL field name
EOFREQUIRED	Reader waits for EOF
IDLE	No-operation wait time
PIPEDEPTH	Buffer number
WAIT	Input-Output wait time
WAITALLOC	Synchronization wait time
WAITCLOSE	Synchronization wait time
WAITEOF	Input-Output wait time
WAITOPEN	Synchronization wait time
WAITTERM	Synchronization wait time

BatchPipes to Job Optimizer Pipes Conversion

Conversion from BatchPipes to Job Optimizer Pipes involves defining Pipe Rules for jobs in which BatchPipes is being replaced by Job Optimizer Pipes. This conversion can be done gradually. First, rules can be defined for new jobs that are implementing Job Optimizer Pipes. Later, existing jobs can be changed to use rules instead of BatchPipes subsystem parameters.

The first step of the conversion should be to review the Pipe default rule (member JOPDEFRL) and check/modify as necessary. This ensures that correct defaults will be used for all pipes. If the conversion is done gradually while using BatchPipes-Compatible Pipes, the fields used for BatchPipes-Compatible Pipes (shown in Table E-3) should not be changed. These values may be re-adjusted only when full conversion is done.

Table E-4 contains information on how to convert each BatchPipes subsystem parameter to a Job Optimizer Pipes rule field. Read the description of the pipe rule fields in Chapter 9, “Job Optimizer Pipes Dialog,” to understand the differences that may arise from the conversion. See Chapter 12, “Job Optimizer Pipes Implementation Considerations,” for more information.

BMC Software, Inc., Confidential and Proprietary Information

Before starting the conversion, note the following differences between BatchPipes or BatchPipes-Compatible Pipes processing and Job Optimizer Pipes pipes processing:

- In BatchPipes, the participant type is decided according to the Open type (input - reader; output - writer). The JCL DISP parameter is not used. However, in Job Optimizer Pipes pipes the participant type is decided according to the PRTYPE subsystem parameter, if specified, the JCL DISP parameter and the pipe rule definition (if the job was specifically defined in the rule). See Chapter 12, “Job Optimizer Pipes Implementation Considerations,” for more information on Job Optimizer Pipes participant type processing.

- In Job Optimizer Pipes, the number of participants allowed to access the pipe is controlled by the minimum writers, additional writers, minimum readers, and additional readers fields. Only the number of participants specified in these fields are allowed to access the pipe (if Additional writers or Additional readers is Y, any number of participants may access the pipe). This is different in BatchPipes processing, where any number of participants can access the pipe while it is active. The Accept additional writers and Accept additional readers initialization parameters do not control the number of participants for Job Optimizer Pipes pipes. It affects only BatchPipes-Compatible Pipes.

- DCB parameters support for Job Optimizer Pipes pipes:

  — BUFNO and EROPT parameters are not used for Job Optimizer Pipes pipes processing. If used for BatchPipes processing, the appropriate Pipe Rule field should be set. Use Table E-2 on page E-13 and Table E-4 on page E-17 to check which fields should be set and how to interpret the values.

  — The LRECL, RECFM and BLKSIZE parameters specified on the DD statement are used as the pipe attributes, unless overridden during Open. This is different than BatchPipes support. In BatchPipes if these parameters are specified on the DD statement they are used as the pipe attributes even if overridden during Open. See ““Record Format” on page 9-31, “Block Size” on page 9-32, and “Logical Record Length” on page 9-33 in Chapter 9, “Job Optimizer Pipes Dialog,” for a full description of DCB attributes for Job Optimizer Pipes pipes.

- Open processing differences:
  - In BatchPipes, a reader can close and re-open the pipe several times during its' process. In Job Optimizer Pipes, a reader can re-open the pipe only if it did not read anything from the pipe. If the reader opens the pipe, reads data, closes and opens again, it will fail.
  - In BatchPipes, a participant is allowed to close the pipe and re-open as a different participant type, (for example, a writer can write to the pipe, close, and open as a reader). This is not allowed in Job Optimizer Pipes. If a participant wants to connect to the pipe with a different type, it has to deallocate the pipe and allocate again.
- There are several Pipe Rule fields which have no equivalent in BatchPipes subsystem parameters. These fields offer features that do not exist in BatchPipes and may be considered when converting from BatchPipes to Job Optimizer Pipes pipes. These parameters are:
  - **Reader data -** This field defines which portion of the data the reader will read. This field affects the processing of pipes having multiple readers:
    - N - (the default). Indicates that each reader will read the next available data block from the pipe. This is how BatchPipes works.
    - A - indicates that all the readers of the pipe will read all the data from the pipe. There is no equivalent BatchPipes parameter. To achieve this functionality in BatchPipes, a fitting has to be created by the pipe writer or reader, to write the data to another pipe and this pipe is read by another reader that needs to read all the pipe data. If there are jobs using this method to enable multiple readers to read all the data, they can be changed to access the same pipe and the Reader data field should be set to A.
  - **Writer signals EOF = Deallocation -** This definition allows the writer to open and close the pipe several times during its' process without writing end-of-file. The end-of-file will be written when the writer finishes its' process and deallocates the pipe. The reader will receive the end-of-file indication after the writer finishes writing all the data.
  - **Reader error option = Delay -** When Job Optimizer Pipes encounters an error in the pipe processing, it will allow the reader to continue reading from the pipe until end-of-file is received. Then the reader will abend. This option prevents losing data written to the pipe, but still indicates that there was a problem in the process.

Table E-4 Converting BatchPipes Parameters to Job Optimizer Pipes Pipe Rules (Part 1 of 3)

BatchPipes Parameter	Job Optimizer Pipes Rule Definition Field Name	Where to Find	Comments
ALLOCNOW	Writer skip alloc synch Reader skip alloc synch	page 9-43 page 9-48	—
ALLOCSYNC	Allocation Synchronization + Minimum readers + Minimum writers	page 9-35 page 9-45 page 9-39	In BatchPipe-Compatible pipes, the minimum number of participants required for allocation synchronization may differ from the number of participants required for open synchronization. In Job Optimizer Pipes pipes, both require the same minimum number of participants for synchronization. Check the values specified for ALLOCSYNC and OPENSYNC and decide the desired values of Minimum Readers, Additional readers, Minimum writers, and Additional writers rule fields.
CLOSESYNC	Close synchronization	page 9-36	—
EOFREQUIRED	Reader waits for EOF	page 9-47	—
ERC	—		There is no equivalent rule field for this parameter. See Chapter 12, “Job Optimizer Pipes Implementation Considerations,” for more information.
ERRPROP	Writer error option Reader error option	page 9-41 page 9-46	In BatchPipes, this parameter defines the error propagation processing for a specific participant. The equivalent Job Optimizer Pipes rule fields define the error processing methods for the writers and readers (one for each type). Check the specification of this parameter in all writers and readers of the pipe and determine the desired values for the reader error option and writer error option rule fields.
FIT/FITDD	Create file + Create file DD name (optional)	page 9-40 page 9-41	A file can be created by the writer only, when there is only one writer that is writing to the pipe. Only one file can be created with the pipe. There is no equivalent for a BatchPipes fitting created by a reader or by a writer when there are multiple writers to a pipe or for multiple fitting created by a participant.

Table E-4 Converting BatchPipes Parameters to Job Optimizer Pipes Pipe Rules (Part 2 of 3)

BatchPipes Parameter	Job Optimizer Pipes Rule Definition Field Name	Where to Find	Comments
IDLE	No-operation wait time+ No-operation wait action	page 9-37 page 9-37	These rule fields define the no-operation time interval for all the participants of the pipe, and the action to be taken when the interval expires. Check the specification of the IDLE BatchPipes parameter in all the participants of the pipe and use the highest value for the No-operation wait time field. Choose a value for the No-operation wait action field. If OFF was specified for IDLE for any of the participants, set 9999 for No-operation wait time and O for No-operation wait action.
NOEOF	Writer signals EOF	page 9-43	—
OPENNOW	Writer skip open synch Reader skip open synch	page 9-44 page 9-49	—
OPENSYNC	Open synchronization + Minimum readers + Minimum writers	page 9-35 page 9-45 page 9-39	See the comment for ALLOCSYNC
PIPEDEPTH	Buffer number	page 9-31	—
TERMSYNC	Deallocation synchronization + (optional) Reader error condition code or Writer error condition code	page 9-36 page 9-46 page 9-43	—
WAIT	Input-Output wait time + Input-Output wait action	page 9-37 page 9-38	These rule fields define the maximum time (in minutes) to wait for data to be transferred to or from the pipe and the action to be taken when data transfer does not occur within the specified time limit. BatchPipes parameters, WAIT and WAITEOF, are converted to Job Optimizer Pipes rule field Input-Output wait time. Check the specification of these BatchPipes parameters in all the participants of the pipe and use the highest value for the Input-Output wait time field. Choose the Input-Output wait action value. If OFF was specified for any of these BatchPipes parameters, set 9999 for Input-Output wait time and O for Input-Output wait action.

Table E-4 Converting BatchPipes Parameters to Job Optimizer Pipes Pipe Rules (Part 3 of 3)

BatchPipes Parameter	Job Optimizer Pipes Rule Definition Field Name	Where to Find	Comments
WAITALLOC	Synchronization wait time + Synchronization wait action	page 9-38 page 9-38	These rule fields define the synchronization time interval for all the participants of the pipe, and the action to be taken when the interval expires. BatchPipes parameters WAITALLOC, WAITOPEN, WAITCLOSE, and WAITTERM are converted to Job Optimizer Pipes rule field Synchronization wait time. Check the specification of these BatchPipes parameters in all the participants of the pipe and use the highest value for the Synchronization wait time field. Choose the Synchronization wait action value. If OFF was specified for any of these BatchPipes parameters, set 9999 for Synchronization wait time and O for Synchronization wait action.
WAITCLOSE	Synchronization wait time + Synchronization wait action	page 9-38 page 9-38	See the comment for WAITALLOC.
WAITEOF	Input-Output wait time + Input-Output wait action	page 9-37 page 9-38	See the comment for WAIT
WAITOPEN	Synchronization wait time + Synchronization wait action	page 9-38 page 9-38	See the comment for WAITALLOC
WAITTERM	Synchronization wait time + Synchronization wait action	page 9-38 page 9-38	See the comment for WAITALLOC

BMC Software, Inc., Confidential and Proprietary Information

Appendix F IOA and INCONTROL User Considerations

This appendix explains the interface between IOA and INCONTROL® components and MAINVIEW® Batch Optimizer and issues that you must consider during installation/customization. This appendix discusses the following topics:

BMC Software, Inc., Confidential and Proprietary Information

Job Optimizer Pipes Setting

When setting the Job Optimizer Pipes initialization parameters, the parameter Group Name is used for the communication between CONTROL-M and Batch Optimizer components.

Job Optimizer Pipes and CONTROL-M use the group name as a token to communicate between the two. The group name is also used to connect together several instances of Job Optimizer Pipes (of the same installation) running on the same image. The Group Name value must be the same as the value specified for the IOA QNAME as set during IOA Installation. See *INCONTROL® for OS/390 Installation Guide* for more information.

Values This option has the following possible values:

Group name 1 - 8 character name.

Default MBOPGRP

Keyword GRPNAME={1-8 character name}

CONTROL-M Setting

Starting with CONTROL-M Release 5.1.4, there is an interface between CONTROL-M and MAINVIEW Batch Optimizer components. For more information about this interface, see “IOA Online Facility” on page F-3. To establish the communication, CONTROL-M installation parameter MVBO should be set to Y.

CONTROL-M Release 5.1.4

Edit member CTMPARM in the INSTCTM library, and add the following line:

```
MVBO=Y
```

Then run job CTMPARMJ (in the same library) to recompile CTMPARM.

BMC Software, Inc., Confidential and Proprietary Information

CONTROL-M Version 6.0.0 or later

This definition is performed under ICE, option 6 (Customize) for product ID CTM. Select option 1 (CTMPARM Post-Installation), and then Step 9 (General Parameters). Variable MVBO should be set to Y. Select Step 10 (Save Parameters into Product Libraries) and restart CONTROL-M.

IOA Online Facility

The IOA online facility enables access to Batch Optimizer information directly from the CONTROL-M Active Environment (Job Status) screen. Option W (when used for an executing job) displays split steps information retrieved from Job Optimizer (when the job contains parallel steps) and pipes information retrieved from Job Optimizer Pipes (when the selected job is a pipe participant).

The displayed information can be limited to only parallel steps or only pipes. In addition, the displayed information can be limited to entries which begin with a specified prefix, run on a specific system, and so forth. The amount of information displayed per entry can also be controlled.

Display Types

The display type controls the amount of information that is displayed for each entry in the screen. While in the screen, the display type can be changed using the DISPLAY command. The DISPLAY command format is the following:

```
DISPLAY x
```

The variable *x* identifies the desired type.

Values This option has the following possible values:

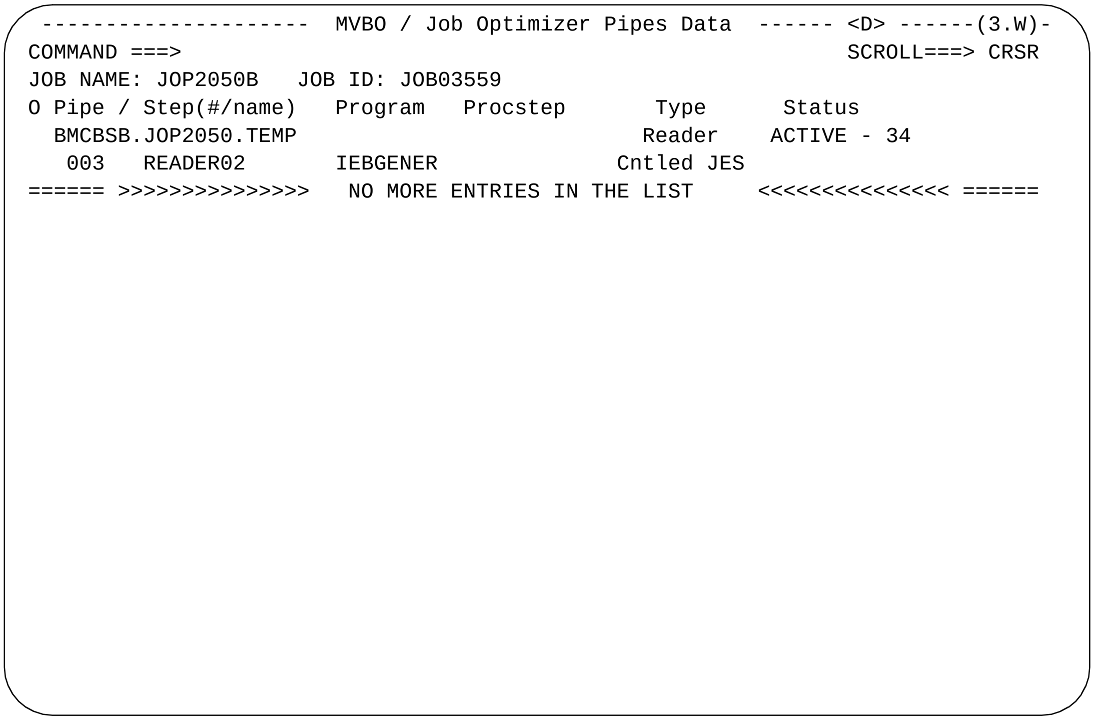
D	minimum information.
A	additional information
S	system programmer (all fields)

Default D

Minimum Information (Display Type D)

Figure F-1 is an example of display type D (minimum information) for pipes and job step entries. Table F-1 lists the Display D fields.

Figure F-1 Minimum Information (Display Type D)



```
 ---------------------  MVBO / Job Optimizer Pipes Data  ------ <D> ------(3.W)-
COMMAND ===>                                                    SCROLL===> CRSR
JOB NAME: JOP2050B   JOB ID: JOB03559
O Pipe / Step(#/name)   Program   Procstep       Type      Status
  BMCBSB.JOP2050.TEMP                           Reader    ACTIVE - 34
   003   READER02       IEBGENER              Cntled JES
====== >>>>>>>>>>>>>>>   NO MORE ENTRIES IN THE LIST     <<<<<<<<<<<<<<< ======
```

Table F-1 Minimum Information (Display Type D) Fields (Part 1 of 2)

Type of Fields	Field	Description
Job-related fields	Job Name	name of the displayed job
	Job ID	job ID of the displayed job
Pipe-related fields	Pipe	pipe name
	Type	participant type - how the participant accesses the pipe (see Table F-2 “Participant Type Values”)
	Status	Pipe status (see Table F-3 “Pipe Status Values”)

Table F-1 Minimum Information (Display Type D) Fields (Part 2 of 2)

Type of Fields	Field	Description
Parallel steps-related fields	Step #	original step number within the job
	Step Name	name of the step that invokes the program
	Program	name of the program that is executed in the step
	Procstep	name of the step that invokes the JCL procedure
	type	type of step (see Table F-4 on page F-6)
	status	step status (see Table F-5 on page F-7)

Participant Type Values

Table F-2 lists the possible participant type values.

Table F-2 Participant Type Values

Value	Description
READER	participant reads data from the pipe
WRITER	participant writes data to the pipe
UNKNWN	participant type is unknown because the participant did not open the pipe yet.

Pipe Status Values

Table F-3 lists possible status values for a pipe.

Table F-3 Pipe Status Values

Value	Description
ALOCSYNC	pipe not fully defined Synchronization type ALLOC was requested but not all of the required participants have allocated the pipe.
OPENSYNC	pipe not fully initialized Synchronization type OPEN was requested but not all of the required participants have opened the pipe.
WAITFWTR	readers which already opened the pipe are waiting for the first writer to open and initialize the pipe Synchronization at open was not requested.
INIT	pipe initialization in process
ACTIVE	pipe is active Data can be transferred into/from the pipe.
AFTEOF	pipe is empty and is marked with EOF indication No more data can be read or written.
CLSSYNC	some of the participants closed the pipe and are waiting for the remaining participants to close Synchronization at CLOSE was requested.
AFTCLOSE	all the participants already closed the pipe
DALCSYNC	some of the participants deallocated the pipe and are waiting for the remaining participants to deallocate Synchronization at DEALLOC was requested.

Note: If the pipe is in error status, the suffix, -ERR, is added to the status. For example, ACTIVE-ERR.

Step Type Values

Table F-4 lists the possible type values for a step in a job.

Table F-4 Step Type Values (Part 1 of 2)

Value	Description
None S/B	job is not under Job Optimizer control
Control	BMC Software Primary Subsystem
Performance	MAINVIEW Batch Optimizer Subsystem
Cntled JES	job contains split steps, but this step is executing in the JES initiator
Split Step	this step is executing in an XJS initiator

Table F-4 Step Type Values (Part 2 of 2)

Value	Description
JES Init	one of the following has occurred: • this JES initiator is inactive • the current job is not being intercepted
XJS	this address space is an inactive XJS initiator
Shadow Step	the JES initiator is executing the Job Optimizer step executor that shadows a previously split step
Status	status of the job step

Step Status Values

Table F-5 lists the possible status values for a step in a job.

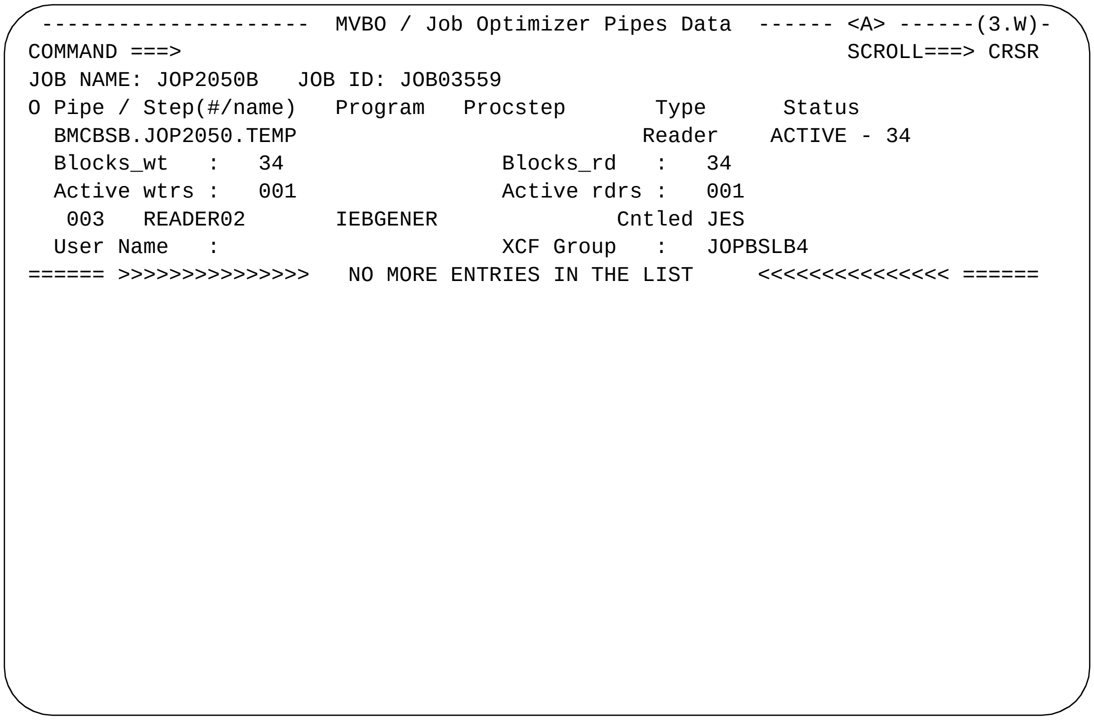
Table F-5 Step Status Values

Value	Description
Pipe Close	step has issued a close for an SRP pipe and is waiting for its partner to respond
EDPL Xfer	a split step must receive all EDPL control blocks prior to initiating
WTR complete	occurs when the writer service task has completed all available work, but Job Optimizer is not ready to execute the next step in the JES initiator
Cancel	Job Optimizer has issued an asynchronous request to cancel this job, but the request has not been serviced
Sysin Data	a split step has issued a READ or GET for SYSIN data, and the SYSIN data has not arrived from the home image
Last Step	the step is waiting for the last split step to finish executing before proceeding
Next Step	the next step to execute in the JES initiator is currently executing in an XJS initiator
Pipe Open	this step has issued an open for an SRP pipe and is waiting for its partner to respond
All Steps	this step is waiting for all prior steps to terminate before continuing
Term Response	waiting for the home image to acknowledge receipt of the split step job log before proceeding
WTR Start	Job Optimizer initiator intercept screen is waiting for the writer service task to initialize

Additional Information (Display Type A)

Figure F-2 is an example of display type A (additional information) for pipes and job step entries. Table F-6 lists additional fields in Display Type A.

Figure F-2 Additional Information (Display Type A)



```
 ---------------------  MVBO / Job Optimizer Pipes Data  ------ <A> ------(3.W)-
COMMAND ===>                                                    SCROLL===> CRSR
JOB NAME: JOP2050B   JOB ID: JOB03559
O Pipe / Step(#/name)   Program   Procstep       Type      Status
  BMCBSB.JOP2050.TEMP                           Reader    ACTIVE - 34
  Blocks_wt   :   34                 Blocks_rd   :   34
  Active wtrs :   001                Active rdrs :   001
   003   READER02       IEBGENER              Cntled JES
  User Name   :                      XCF Group   :   JOPBSLB4
====== >>>>>>>>>>>>>>>   NO MORE ENTRIES IN THE LIST     <<<<<<<<<<<<<<< ======
```

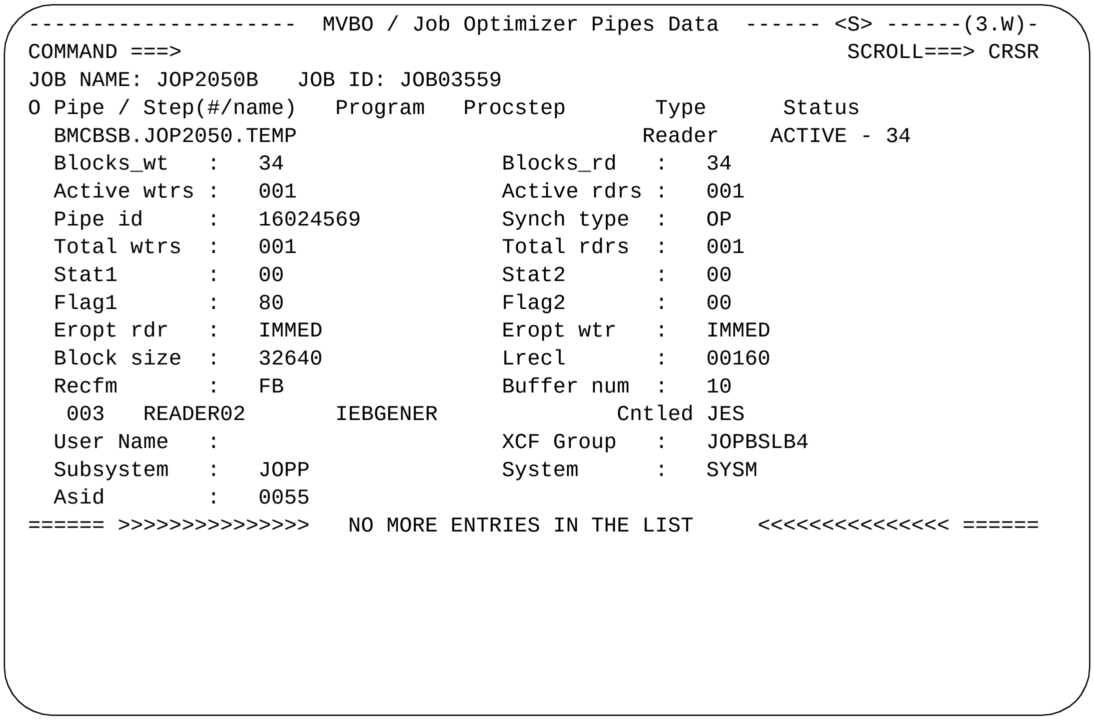
Table F-6 Additional fields in Display Type A

Type of Fields	Field	Description
Pipe-related fields	Blocks_wt	total number of blocks written by all the writers
	Blocks_rd	total number of blocks read by all the readers
	Active wtrs	number of writers that are currently connected to the pipe
	Active rdrs	number of readers that are currently connected to the pipe
Parallel steps-related fields	User Name	reserved for future use
	XCF Group	contains the XCF Group name used by Job Optimizer

System Programmer (Display Type S)

Figure F-3 is an example of display type S (system programmer) for pipes and job step entries. Table F-7 on page F-9 lists additional fields in Display Type S.

Figure F-3 System Programmer (Display Type S)



```
---------------------  MVBO / Job Optimizer Pipes Data  ------ <S> ------(3.W)-
COMMAND ===>                                                    SCROLL===> CRSR
JOB NAME: JOP2050B   JOB ID: JOB03559
O Pipe / Step(#/name)   Program   Procstep       Type      Status
  BMCBSB.JOP2050.TEMP                           Reader    ACTIVE - 34
  Blocks_wt   :   34                 Blocks_rd   :   34
  Active wtrs :   001                Active rdrs :   001
  Pipe id     :   16024569           Synch type  :   OP
  Total wtrs  :   001                Total rdrs  :   001
  Stat1       :   00                 Stat2       :   00
  Flag1       :   80                 Flag2       :   00
  Eropt rdr   :   IMMED              Eropt wtr   :   IMMED
  Block size  :   32640              Lrecl       :   00160
  Recfm       :   FB                 Buffer num  :   10
   003   READER02       IEBGENER              Cntled JES
  User Name   :                      XCF Group   :   JOPBSLB4
  Subsystem   :   JOPP               System      :   SYSM
  Asid        :   0055
====== >>>>>>>>>>>>>>>   NO MORE ENTRIES IN THE LIST     <<<<<<<<<<<<<<< ======
```

Table F-7 Additional fields in Display Type S (Part 1 of 2)

Type of Fields	Field	Description
Pipe-related fields	Pipe id	unique pipe identifier
	Synch type	synchronization points requested for the pipe. Valid values: NONE (no synchronization), or any combination of: ALLOC, OPEN, CLOSE and DEALLOC
	Total wtrs	total number of writers that are or were connected to the pipe
	Total rdrs	total number of readers that are or were connected to the pipe
	Stat 1	internal status field in hex format
	Stat 2	internal status field in hex format
	Flag 1	internal status field in hex format
	Flag 2	internal status field in hex format
	Eropt rdr	reader error option
	Eropt wtr	writer error option
	Block size	pipe block size
	Lrecl	logical record size
	Recfm	record format
	Buffer num	number of buffers

Table F-7 Additional fields in Display Type S (Part 2 of 2)

Type of Fields	Field	Description
Parallel steps-related fields	Subsystem	name of Job Optimizer subsystem
	System	name of the system where the first step executed. This is the home image
	Asid	address-space ID of the selected job

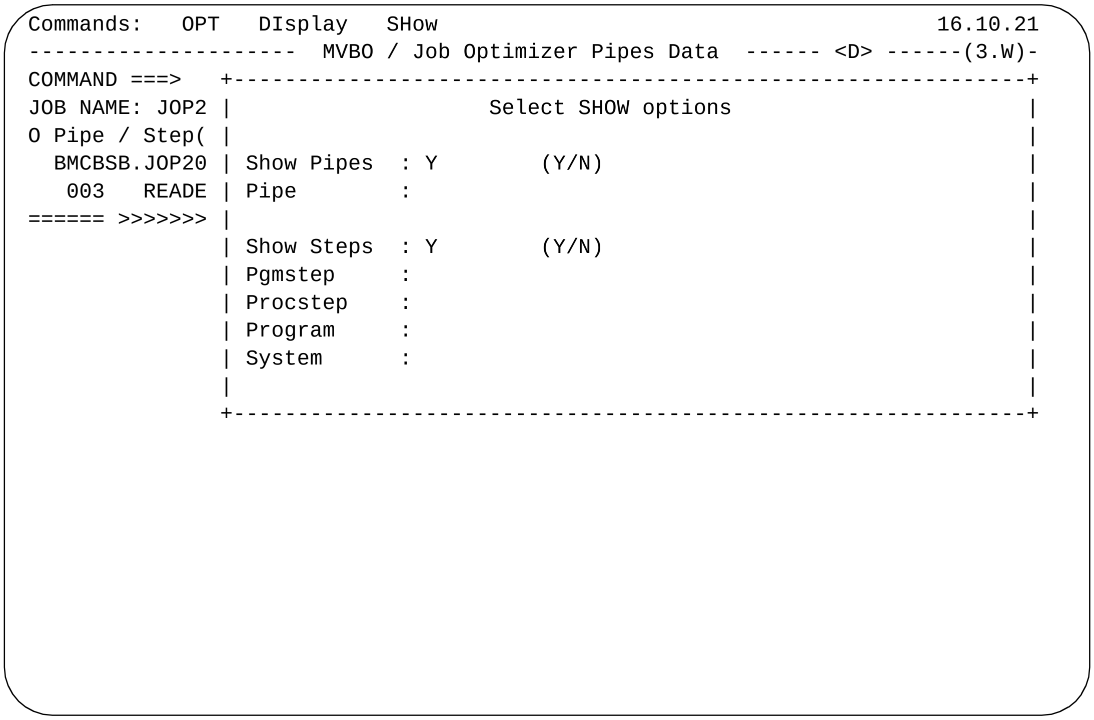
The following commands can be specified in the COMMAND field.

OPT	Display/Hide options list
DISPLAY x	Modify the display type
SHOW	Open the Selection Criteria panel (shown in Figure F-4 on page F-11)

Selection Criteria Panel

The displayed list of pipes and steps can be limited to include only entries with certain attributes. The SHOW command opens a window that enables selection criteria for entries. Figure F-4 is an example of the Selection Criteria panel. Figure F-4 lists the selection criteria panel fields.

Figure F-4 Selection Criteria Panel



```
Commands:   OPT   DIsplay   SHow                                      16.10.21
---------------------  MVBO / Job Optimizer Pipes Data  ------ <D> ------(3.W)-
COMMAND ===>   +--------------------------------------------------------------+
JOB NAME: JOP2 |                    Select SHOW options                       |
O Pipe / Step( |                                                              |
  BMCBSB.JOP20 | Show Pipes  : Y        (Y/N)                                 |
   003   READE | Pipe        :                                                |
====== >>>>>>> |                                                              |
               | Show Steps  : Y        (Y/N)                                 |
               | Pgmstep     :                                                |
               | Procstep    :                                                |
               | Program     :                                                |
               | System      :                                                |
               |                                                              |
               +--------------------------------------------------------------+
```

Table F-8 Selection Criteria Panel Fields (Part 1 of 2)

Field	Description
Show Pipes	determines whether pipe entries will be displayed valid values: • Y (default) • N
Pipe	mask string used for selecting pipes by the pipe name valid values: • 1-44 alphanumeric characters • mask characters * and ? are allowed
Show Steps	determines whether step entries will be displayed valid values: • Y (default) • N
Pgmstep	mask string used for selecting steps by the step name valid values: • 1-8 alphanumeric characters • mask characters * and ? are allowed
Procstep	mask string used for selecting steps by the step name that executes a JCL procedure valid values: • 1-8 alphanumeric characters • mask characters * and ? are allowed

Table F-8 Selection Criteria Panel Fields (Part 2 of 2)

Field	Description
Program	mask string used for selecting steps by the executed program name. valid values: • 1-8 alphanumeric characters • mask characters * and ? are allowed
System	mask string used for selecting steps by the system where they execute valid values: • 1-8 alphanumeric characters • mask characters * and ? are allowed

Implementation Considerations

This section discusses implementation issues to be considered for IOA and INCONTROL® products support for MAINVIEW Batch Optimizer.

CONTROL-M Scheduling and Control

This section discusses batch jobs parallel processing and parallel steps processing for CONTROL-M Scheduling and Control.

Batch Jobs Parallel Processing

The job-to-job pipe mechanism is based on efficient, parallel processing of pipe participants. To achieve this goal, CONTROL-M has been enhanced to enable concurrent submission of pipe participants.

CONTROL-M recognizes and treats the set of interrelated pipes and participants as a single, comprehensive unit, called a collection.

Because CONTROL-M uses a collection as the basic unit of work to be scheduled, all pipe participants are submitted concurrently, after verification that all required resources (that is, prerequisite conditions, quantitative resources) are available.

This method ensures that some participants will not wait for other participants (that is, at synchronization points) to start executing.

Example

Assume the following situation:

- JOB_A writes to pipe P1.
- JOB_B reads from pipe P1 and writes to pipe P2.
- JOB_C reads from pipe P2.

Pipe P1 is used by JOB_A and JOB_B. Pipe P2 is used by JOB_B and JOB_C. The collection consists of 3 jobs, JOB_A, JOB_B and JOB_C, which use both pipes (P1 and P2).

CONTROL-M scheduling support consists of the following components:

- Job Scheduling Definition Support
- Enhanced Runtime Scheduling Algorithm

Note: Additional information for CONTROL-M Scheduling and Control support can be found in the *CONTROL-M for OS/390 User Manual*. Within the *CONTROL-M for OS/390 User Manual*, all references to CONTROL-M/WorkLoad should be replaced with references to MAINVIEW Batch Optimizer.

Job Scheduling Definition Support

The CONTROL-M job scheduling definition has been enhanced to contain PIPE statements for each pipe accessed by a job. Each PIPE statement contains the pipe (dataset) name. A participant's job scheduling definition will include a PIPE statement for each pipe accessed by the job. When scheduling the tables, CONTROL-M identifies the pipe participants and arranges them in a collection.

Enhanced Runtime Scheduling Algorithm

When jobs that are part of a collection are scheduled, CONTROL-M treats the whole collection as one unit of work for processing runtime scheduling criteria, as follows:

CONTROL-M ensures that Job Optimizer Pipes is active and that the required number of participants access each pipe in the collection. The required (minimum) number of participants is obtained from Job Optimizer Pipes rules which are active at that time. If a rule is not found (which will be the case for pipe defined in JCL), a minimum of one writer and one reader is required. If a participant is missing, the jobs in the collection are not submitted. This method ensures that a job is not submitted when its participants are not scheduled on that day.

CONTROL-M analyzes all resources (that is, prerequisite conditions, quantitative resources, time limits) required by all jobs in the collection as a single unit. All participants are submitted together when all the resources required by the collection are available. This ensures the parallel submission of pipe participants. CONTROL-M ignores prerequisite conditions between all jobs in the collection. Without using pipes, the jobs are run sequentially, usually by having prerequisite conditions set between them. In order to run these jobs in parallel, those prerequisite conditions are ignored.

To ensure that jobs will start execution on time, it is recommended that initiators be handled as quantitative resources. This ensures that submitted jobs do not wait for initiators and hold up the other jobs in the collection.

When all the jobs in the collection are ready to be submitted, CONTROL-M ensures that the Job Optimizer Pipes address space is active. If the Job Optimizer Pipes address space is not active, the jobs in the collection are not submitted. This ensures that the jobs can be run in parallel using pipes.

Parallel Steps Processing

CONTROL-M handles a job that will be split by Job Optimizer in the same manner it handles a sequential job. The scheduling process is similar to scheduling any batch job. Only when the job starts execution in the system, Job Optimizer decides whether it will be split or not. Therefore, CONTROL-M only monitors the original job. It has no control over the parallel steps which are executed outside of the original job. During the job execution, it is possible to monitor the job steps using option 3.W from the CONTROL-M Active Environment screen. The displayed information is supplied by Job Optimizer, which monitors each step of the job.

CONTROL-M/R Rerun/Restart Considerations

This section discussed batch jobs parallel processing and parallel jobs processing for CONTROL-M/R Rerun/Restart Considerations.

Batch Jobs Parallel Processing

When a participant fails, Job Optimizer Pipes notifies all the other participants about the error. The participants either abend or continue, according to the Error Options defined for the pipe. (For more information, see “Error Handling and Distribution” on page 12-26.) Therefore, depending on the Error Options, restart can be required for either the entire collection or for a subset of the collection. For more information, see “Restart Considerations” on page 12-32.

- If the entire collection should be restarted, restart should be performed for each of the jobs in the collection.
- If only a subset of the collection is to be restarted (for example, the participants of only one pipe), the PIPE statements should be adjusted (in screen 3. Zoom) to form a collection containing only the desired jobs. Restart should then be performed for these jobs.
- If only one job should be restarted (for example, a reader that should read the physical file created by the writer), the PIPE statement should be removed from that job (in screen 3.zoom), and the pipe rule should be held or deleted before the job is restarted. If the pipe was defined in JCL, the JCL of the job should be modified to use a file instead of a pipe before the job is restarted.

Parallel Steps Processing

CONTROL-R automatically supports split jobs. The restart will always begin from the step that failed (or earlier if required) and include all subsequent steps. For example, steps 2 and 4 were running in parallel. Step 4 ended OK and step 2 failed. The restart will start from step 2 (the step that failed) and will also execute again step 4 (although it ended OK). This is the same situation that occurs if certain steps of job contain COND=EVEN, which causes the step to run even if a prior step abended.

There are special situations that are relevant only for split jobs. For example, steps 2 and 3 of a job were running in parallel, and step 3 completed execution while step 2 was still running. If a system failure occurs and an IPL is performed while step 2 is running, the output of step 3 is not merged into the original job. When analyzing the output of the job, CONTROL-R will be able to determine that step 2 failed, but will not get any information regarding the execution of step 3.

CMEM Considerations

This section discusses batch jobs processing and parallel steps processing for CMEM considerations.

Batch Jobs Parallel Processing and Parallel Steps Processing

When a dataset is replaced by a pipe, the pipe cannot trigger DSNEVENT rules because there is no dataset. (The single exception to this is described below.) Therefore, checking DSNEVENT rules against pipes definition (either via rules or in JCL) enables you to determine which DSNEVENT rules do not get triggered.

When checking DSNEVENT rules, the following should be considered:

- Many DSNEVENT rules are designed to FORCE a job or add a condition to run a job that will read the file just created. If the file is replaced by a pipe, and the jobs triggered by this rule become pipe participants, this rule is not needed. However, we recommend leaving the rule to support situations in which the jobs do not use a pipe.

- When DSNEVENT rules reference a dataset that is replaced by a pipe, but the rule action is still required, consider replacing the DSNEVENT with a STEP event. In this way, the action will not depend on the dataset but on the step end event and/or on the completion code of the step. This method is usable if the DSNEVENT actions are performed when the step ends. For historical reasons, many DSNEVENT rules are actually rules intended to activate another process when a step ends. Such rules can be replaced by STEP rules.

- When data sets are referenced by DSNEVENT rules that cannot be replaced with STEP event rules and the actions defined in those DSNEVENT rules are still required, pipes cannot be used.

The exception to this scenario is when a physical file is created along with the pipe. For these files, DSNEVENT rules with DISP=CATLG/RETAIN will be triggered when the writer creates the file. Such rules should be checked to determine whether or not they are required, and whether or not the action they perform, if they remain active, is acceptable when a pipe is used.

In this case, the reader does not access the physical file. Therefore, DSNEVENT rules defined to trigger this access will not be triggered and their actions will not be performed.

CONTROL-O Considerations

CONTROL-O rules are triggered by events occurring in the system. Pipes and parallel steps processing may change these events, the number of times they occur or their sequence.

Batch Jobs Parallel Processing and Parallel Steps Processing

For dataset replaced by pipes, see “CMEM Considerations” on page F-16.

CONTROL-D Decollation Considerations

CONTROL-D decollation can decollate one or more sysouts of the same job. The selection of the required sysout(s) may be done using the sysout DDNAME. Sometimes, the order of the decollated sysouts is important for the sysout distribution. If the job that produces the sysout is split by Job Optimizer, Job Policy parameter Spool Scheduling Option determines how the steps sysouts will be gathered back to the job. Specifying the value of DEFER in the Spool Scheduling Option tells Job Optimizer to gather the sysouts in the same manner as for normal sequential processing and to retain their DDNAMEs. For more information, see “Spool Scheduling Option” on page 4-27.

CONTROL-D uses the step number as part of the dataset name when allocating files. Use the STEPNUM ON setting to force Job Optimizer to set a split step to its original step number. For more information, see “Controlling the SCT with Job Optimizer” on page 6-37.

Appendix G CA-11 Considerations

The Job Optimizer component requires the CA-11 exit U11ACTEX. CA-11 tracks the shadow steps in the original job, not the steps that are executing in XJS initiators. The U11ACTEX exit that is supplied by BMC Software identifies the steps executing in XJS initiators and disables CA-11 tracking for those steps.

This exit will not impede the CA-11 ability to perform restart against a job under Job Optimizer control. The U11ACTEX exit will not affect the tracking of jobs that are not under Job Optimizer control.

Note: The following procedure is based on CA-11 version 2 release 2. If this release is not your release of CA-11, contact BMC Software Customer Support at 800 841 2031.

To install CA-11 USERMOD U11ACTEX:

Step 1 Edit the sample JCL in the CA-11 SAMPJCL library member L722UXIT, making the following modifications:

1.A Change the JOBCARD according to your site’s standards.

1.B Change the SMPPTFIN DD statement as follows, changing BSS.SAMPLIB to reflect the name of your MAINVIEW Batch Optimizer sample library:

```
// SMPE.SMPPTFIN DD DSN=BMC.BSS.SAMPLIB(U11ACT00),
// DISP=SHR,UNIT=,VOL=,LABEL=
```

1.C Change the SMPPTFIN DD statement as follows, changing BSS.SAMPLIB to reflect the name of your MAINVIEW Batch Optimizer sample library:

```
//SMPCNTL  DD DSN=BMC.BSS.SAMPLIB(U11SMCTL),DISP=SHR
```

Step 2 Review BMC.BSS.SAMPLIB member U11SMCTL and verify that the specified target zone (CA11TGT) matches the CA-11 target zone at your site. (If not, change CA11TGT to the name of your CA-11 target zone.)

Step 3 Submit the job that was edited in Step 1. A return code less than or equal to 4 is acceptable. If you received a return code greater than 4, review the job output and correct the errors. If no errors are apparent, contact BMC Software Customer Support at 800 841 2031.

Step 4 Change the CA-11 DBAS parameter deck to include the following content:

```
RECOVCSA=NO
```

This change forces CA-11 to reload the user exit when the DBAS is restarted.

Step 5 Restart the DBAS.

Installation of the U11ACTEX exit required by MAINVIEW Batch Optimizer is complete.

Note: You can verify that the exit is active by tracking a job with CA-11 while Job Optimizer is splitting steps. For the steps that split, you will find the following message:

U11-610 JOBNAME=*xxxxxx*,ACTION EXIT (U11ACTEX) DECIDED NOT TO TRACK

In the preceding message, *xxxxxx* is the job name that you want to run.

Index

BMC Software, Inc., Confidential and Proprietary Information

END USER LICENSE AGREEMENT
NOTICE

BY OPENING THE PACKAGE, INSTALLING, PRESSING "AGREE" OR "YES" OR USING THE PRODUCT, THE ENTITY OR INDIVIDUAL ENTERING INTO THIS AGREEMENT AGREES TO BE BOUND BY THE FOLLOWING TERMS. IF YOU DO NOT AGREE WITH ANY OF THESE TERMS, DO NOT INSTALL OR USE THE PRODUCT, PROMPTLY RETURN THE PRODUCT TO BMC OR YOUR BMC RESELLER, AND IF YOU ACQUIRED THE LICENSE WITHIN 30 DAYS OF THE DATE OF YOUR ORDER CONTACT BMC OR YOUR BMC RESELLER FOR A REFUND OF LICENSE FEES PAID. IF YOU REJECT THIS AGREEMENT, YOU WILL NOT ACQUIRE ANY LICENSE TO USE THE PRODUCT.

This Agreement ("**Agreement**") is between the entity or individual entering into this Agreement ("You") and BMC Software Distribution, Inc., a Delaware corporation located at 2101 CityWest Blvd., Houston, Texas, 77042, USA or its affiliated local licensing entity ("BMC"). "You" includes you and your Affiliates. "Affiliate" is defined as an entity which controls, is controlled by or shares common control with a party. IF MORE THAN ONE LICENSE AGREEMENT COULD APPLY TO THE PRODUCT, THE FOLLOWING ORDER OF LICENSE AGREEMENT PRECEDENCE APPLIES: (1) WEB BASED LICENSE AGREEMENT WITH BMC, (2) WRITTEN LICENSE AGREEMENT WITH BMC, (3) SHRINK-WRAP LICENSE AGREEMENT WITH BMC PROVIDED WITH THE PRODUCT, AND (4)THIS ELECTRONIC LICENSE AGREEMENT WITH BMC. In addition to the restrictions imposed under this Agreement, any other usage restrictions contained in the Product installation instructions or release notes shall apply to Your use of the Product.

PRODUCT AND CAPACITY. "Software" means the object code version of the computer programs provided, via delivery or electronic transmission, to You. Software includes computer files, enhancements, maintenance modifications, upgrades, updates, bug fixes, and error corrections.

"Documentation" means all written or graphical material provided by BMC in any medium, including any technical specifications, relating to the functionality or operation of the Software.

"Product" means the Software and Documentation.

"License Capacity" means the licensed capacity for the Software with the pricing and other license defining terms, including capacity restrictions, such as tier limit, total allowed users, gigabyte limit, quantity of Software, and/or other capacity limitations regarding the Software. For licenses based on the power of a computer, You agree to use BMC's current computer classification scheme, which is available at http://www.bmc.com or can be provided to You upon request.

ACCEPTANCE. The Product is deemed accepted by You, on the date that You received the Product from BMC.

LICENSE. Subject to the terms of this Agreement, as well as Your payment of applicable fees, BMC grants You a non-exclusive, non-transferable, perpetual (unless a term license is provided on an order) license for each copy of the Software, up to the License Capacity, to do the following:

(a) install the Software on Your owned or leased hardware located at a facility owned or controlled by You in the country where You acquired the license;

(b) operate the Software solely for processing Your own data in Your business operations; and

(c) make one copy of the Software for backup and archival purposes only (collectively a **"License"**).

If the Software is designed by BMC to permit you to modify such Software, then you agree to only use such modifications or new software programs for Your internal purposes or otherwise consistent with the License. BMC grants You a license to use the Documentation solely for Your internal use in Your operations.

LICENSE UPGRADES. You may expand the scope of the License Capacity only pursuant to a separate agreement with BMC for such expanded usage and Your payment of applicable fees. There is no additional warranty period or free support period for license upgrades.

RESTRICTIONS: You agree to **NOT**:

(a) disassemble, reverse engineer, decompile or otherwise attempt to derive any Software from executable code;

(b) distribute or provide the Software to any third party (including without limitation, use in a service bureau, outsourcing environment, or processing the data of third parties, or for rental, lease, or sublicense); or

(c) provide a third party with the results of any functional evaluation or benchmarking or performance tests, without BMC's prior written approval, unless prohibited by local law.

TRIAL LICENSE. If, as part of the ordering process, the Product is provided on a trial basis, then these terms apply: (i) this license consists solely of a non-exclusive, non-transferable evaluation license to operate the Software for the period of time specified from BMC or, if not specified, a 30 day time period (**"Trial Period"**) only for evaluating whether You desire to acquire a capacity-based license to the Product for a fee; and (ii) Your use of the Product is on an AS IS basis without any warranty, and **BMC, ITS AFFILIATES AND RESELLERS, AND LICENSORS DISCLAIM ANY AND ALL WARRANTIES (INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT) AND HAVE NO LIABILITY WHATSOEVER RESULTING FROM THE USE OF THIS PRODUCT UNDER THIS TRIAL LICENSE ("Trial License")**. BMC may terminate for its convenience a Trial License upon notice to You. When the Trial Period ends, Your right to use this Product automatically expires. If You want to continue Your use of the Product beyond the Trial Period, contact BMC to acquire a capacity-based license to the Product for a fee.

TERMINATION. This Agreement shall immediately terminate if You breach any of its terms. Upon termination, for any reason, You must uninstall the Software, and either certify the destruction of the Product or return it to BMC.

OWNERSHIP OF THE PRODUCT. BMC or its Affiliates or licensors retain all right, title and interest to and in the BMC Product and all intellectual property, informational, industrial property and proprietary rights therein. BMC neither grants nor otherwise transfers any rights of ownership in the BMC Product to You. Products are protected by applicable copyright, trade secret, and industrial and intellectual property laws. BMC reserves any rights not expressly granted to You herein.

CONFIDENTIAL AND PROPRIETARY INFORMATION. The Products are and contain valuable confidential information of BMC (**"Confidential Information"**). Confidential Information means non-public technical and non-technical information relating to the Products and Support, including, without limitation, trade secret and proprietary information, and the structure and organization of the Software. You may not disclose the Confidential Information to third parties. You agree to use all reasonable efforts to prevent the unauthorized use, copying, publication or dissemination of the Product.

WARRANTY. Except for a Trial License, BMC warrants that the Software will perform in substantial accordance with the Documentation for a period of one year from the date of the order. This warranty shall not apply to any problems caused by software or hardware not supplied by BMC or to any misuse of the Software.

EXCLUSIVE REMEDY. BMC's entire liability, and Your exclusive remedy, for any defect in the Software during the warranty period or breach of the warranty above shall be limited to the following: BMC shall use reasonable efforts to remedy defects covered by the warranty or replace the defective Software within a reasonable period of time, or if BMC cannot remedy or replace such defective copy of the Software, then BMC shall refund the amount paid by You for the License for that Software. BMC's obligations in this section are conditioned upon Your providing BMC prompt access to the affected Software and full cooperation in resolving the claim.

DISCLAIMER. EXCEPT FOR THE EXPRESS WARRANTIES ABOVE, THE PRODUCT IS PROVIDED "AS IS." BMC, ITS AFFILIATES AND LICENSORS SPECIFICALLY DISCLAIM ALL OTHER WARRANTIES, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT. BMC DOES NOT WARRANT THAT THE OPERATION OF THE SOFTWARE WILL BE UNINTERRUPTED OR ERROR FREE, OR THAT ALL DEFECTS CAN BE CORRECTED.

DISCLAIMER OF DAMAGES. IN NO EVENT IS BMC, ITS AFFILIATES OR LICENSORS LIABLE FOR ANY SPECIAL, INDIRECT, INCIDENTAL, PUNITIVE OR CONSEQUENTIAL DAMAGES RELATING TO OR ARISING OUT OF THIS AGREEMENT, SUPPORT, AND/OR THE PRODUCT (INCLUDING, WITHOUT LIMITATION, LOST PROFITS, LOST COMPUTER USAGE TIME, AND DAMAGE OR LOSS OF USE OF DATA), EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGES, AND IRRESPECTIVE OF ANY NEGLIGENCE OF BMC OR WHETHER SUCH DAMAGES RESULT FROM A CLAIM ARISING UNDER TORT OR CONTRACT LAW.

LIMITS ON LIABILITY. BMC'S AGGREGATE LIABILITY FOR DAMAGES IS LIMITED TO THE AMOUNT PAID BY YOU FOR THE LICENSE TO THE PRODUCT.

SUPPORT. If Your order includes support for the Software, then BMC agrees to provide support (24 hours a day/7 days a week) (**"Support"**). You will be automatically re-enrolled in Support on an annual basis unless BMC receives notice of termination from You as provided below. There is a free support period during the one year warranty period.

(a) **Support Terms.** BMC agrees to make commercially reasonable efforts to provide the following Support: (i) For malfunctions of supported versions of the Software, BMC provides bug fixes, patches or workarounds in order to cause that copy of the Software to operate in substantial conformity with its then-current operating specifications; and (ii) BMC provides new releases or versions, so long as such new releases or versions are furnished by BMC to all other enrolled Support customers without additional charge. BMC may refuse to provide Support for any versions or releases of the Software other than the most recent version or release of such Software made available by BMC. Either party may terminate Your enrollment in Support upon providing notice to the other at least 30 days prior to the next applicable Support anniversary date. If You re-enroll in Support, BMC may charge You a reinstatement fee of 1.5 times what You would have paid if You were enrolled in Support during that time period.

(b) **Fees.** The annual fee for Support is 20% of the Software's list price less the applicable discount or a flat capacity based annual fee. BMC may change its prices for the Software and/or Support upon at least 30 days notice prior to Your support anniversary date.

VERIFICATION. If requested by BMC, You agree to deliver to BMC periodic written reports, whether generated manually or electronically, detailing Your use of the Software in accordance with this Agreement, including, without limitation, the License Capacity. BMC may, at its expense, perform an audit, at your facilities, of Your use of the Software to confirm Your compliance with the Agreement. If an audit reveals that You have underpaid fees, You agree to pay such underpaid fees. If the underpaid fees exceed 5% of the fees paid, then You agree to also pay BMC's reasonable costs of conducting the audit.

EXPORT CONTROLS. You agree not to import, export, re-export, or transfer, directly or indirectly, any part of the Product or any underlying information or technology except in full compliance with all United States, foreign and other applicable laws and regulations.

GOVERNING LAW. This Agreement is governed by the substantive laws in force, without regard to conflict of laws principles: (a) in the State of New York, if you acquired the License in the United States, Puerto Rico, or any country in Central or South America; (b) in the Province of Ontario, if you acquired the License in Canada (subsections (a) and (b) collectively referred to as the **"Americas Region"**); (c) in Singapore, if you acquired the License in Japan, South Korea, Peoples Republic of China, Special Administrative Region of Hong Kong, Republic of China, Philippines, Indonesia, Malaysia, Singapore, India, Australia, New Zealand, or Thailand (collectively, **"Asia Pacific Region"**); or (d) in the Netherlands, if you acquired the License in any other country not described above. The United Nations Convention on Contracts for the International Sale of Goods is specifically disclaimed in its entirety.

ARBITRATION. ANY DISPUTE BETWEEN YOU AND BMC ARISING OUT OF THIS AGREEMENT OR THE BREACH OR ALLEGED BREACH, SHALL BE DETERMINED BY BINDING ARBITRATION CONDUCTED IN ENGLISH. IF THE DISPUTE IS INITIATED IN THE AMERICAS REGION, THE ARBITRATION SHALL BE HELD IN NEW YORK, U.S.A., UNDER THE CURRENT COMMERCIAL OR INTERNATIONAL, AS APPLICABLE, RULES OF THE AMERICAN ARBITRATION ASSOCIATION. IF THE DISPUTE IS INITIATED IN A COUNTRY IN THE ASIA PACIFIC REGION, THE ARBITRATION SHALL BE HELD IN SINGAPORE, SINGAPORE UNDER THE CURRENT UNCITRAL ARBITRATION RULES. IF THE DISPUTE IS INITIATED IN A COUNTRY OUTSIDE OF THE AMERICAS REGION OR ASIA PACIFIC REGION, THE ARBITRATION SHALL BE HELD IN AMSTERDAM, NETHERLANDS UNDER THE CURRENT UNCITRAL ARBITRATION RULES. THE COSTS OF THE ARBITRATION SHALL BE BORNE EQUALLY PENDING THE ARBITRATOR'S AWARD. THE AWARD RENDERED SHALL BE FINAL AND BINDING UPON THE PARTIES AND SHALL NOT BE SUBJECT TO APPEAL TO ANY COURT, AND MAY BE ENFORCED IN ANY COURT OF COMPETENT JURISDICTION. NOTHING IN THIS AGREEMENT SHALL BE DEEMED AS PREVENTING EITHER PARTY FROM SEEKING INJUNCTIVE RELIEF FROM ANY COURT HAVING JURISDICTION OVER THE PARTIES AND THE SUBJECT MATTER OF

THE DISPUTE AS NECESSARY TO PROTECT EITHER PARTY'S CONFIDENTIAL INFORMATION, OWNERSHIP, OR ANY OTHER PROPRIETARY RIGHTS. ALL ARBITRATION PROCEEDINGS SHALL BE CONDUCTED IN CONFIDENCE, AND THE PARTY PREVAILING IN ARBITRATION SHALL BE ENTITLED TO RECOVER ITS REASONABLE ATTORNEYS' FEES AND NECESSARY COSTS INCURRED RELATED THERETO FROM THE OTHER PARTY.

U.S. GOVERNMENT RESTRICTED RIGHTS. The Software under this Agreement is "commercial computer software" as that term is described in 48 C.F.R. 252.227-7014(a)(1). If acquired by or on behalf of a civilian agency, the U.S. Government acquires this commercial computer software and/or commercial computer software documentation subject to the terms of this Agreement as specified in 48 C.F.R. 12.212 (Computer Software) and 12.211 (Technical Data) of the Federal Acquisition Regulations (**"FAR"**) and its successors. If acquired by or on behalf of any agency within the Department of Defense (**"DOD"**), the U.S. Government acquires this commercial computer software and/or commercial computer software documentation subject to the terms of this Agreement as specified in 48 C.F.R. 227.7202 of the DOD FAR Supplement and its successors.

MISCELLANEOUS TERMS. You agree to pay BMC all amounts owed no later than 30 days from the date of the applicable invoice, unless otherwise provided on the order for the License to the Products. You will pay, or reimburse BMC, for taxes of any kind, including sales, use, duty, tariffs, customs, withholding, property, value-added (VAT), and other similar federal, state or local taxes (other than taxes based on BMC's net income) imposed in connection with the Product and/or the Support. This Agreement constitutes the entire agreement between You and BMC and supersedes any prior or contemporaneous negotiations or agreements, whether oral, written or displayed electronically, concerning the Product and related subject matter. No modification or waiver of any provision hereof will be effective unless made in a writing signed by both BMC and You. You may not assign or transfer this Agreement or a License to a third party without BMC's prior written consent. Should any provision of this Agreement be invalid or unenforceable, the remainder of the provisions will remain in effect. The parties have agreed that this Agreement and the documents related thereto be drawn up in the English language. Les parties exigent que la présente convention ainsi que les documents qui s'y rattachent soient rédigés en anglais.

SW Click EULA 071102

Notes

28298